BEST AVAILABLE IMAGES

Defective images within this document are accurate representations of the original documents submitted by the applicant.

Defects in the images may include (but are not limited to):

- BLACK BORDERS
- TEXT CUT OFF AT TOP, BOTTOM OR SIDES
- FADED TEXT
- ILLEGIBLE TEXT
- SKEWED/SLANTED IMAGES
- COLORED PHOTOS
- BLACK OR VERY BLACK AND WHITE DARK PHOTOS
- GRAY SCALE DOCUMENTS

IMAGES ARE BEST AVAILABLE COPY.

As rescanning documents *will not* correct images, please do not report the images to the Image Problem Mailbox.

(19)日本国特許庁（ＪＰ）　(12)公開特許公報（Ａ）　(11)特許出願公開番号
特開2000－36737
（P2000－36737A）
(43)公開日　平成12年２月２日(2000.2.2)

(51)Int.Cl.'	識別記号	FI		テーマコード'（参考）
H03K 19/173	101	H03K 19/173	101	
G06F 17/50		19/177		
H01L 21/82		G06F 15/60	664P	
H03K 19/177			670D	
		H01L 21/82	A	

審査請求　有　請求項の数13　ＯＬ　（全 85 頁）

(21)出願番号　特願平11－132028
(62)分割の表示　特願平1－509588の分割
(22)出願日　平成１年10月４日(1989.10.4)

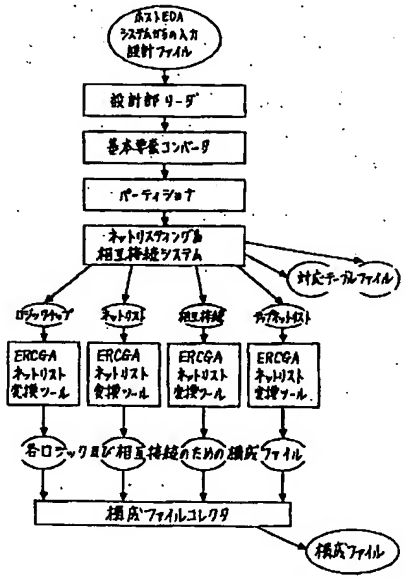
(31)優先権主張番号　２５４４６３
(32)優先日　昭和63年10月５日(1988.10.5)
(33)優先権主張国　米国（ＵＳ）

(71)出願人　595043376
クイックターン・デザイン・システムズ・インコーポレイテッド
QUICKTURN DESIGN SYSTEMS, INC.
アメリカ合衆国95131カリフォルニア州サンノゼ、ウエスト・トリンブル・ロード55番
(72)発明者　パツツ　マイケル　アール
アメリカ合衆国オレゴン州　97212　ポートランド　エヌ　イー　アラミーダ　3336
(74)代理人　100064344
弁理士　岡田　英彦　（外３名）

最終頁に続く

(54)【発明の名称】　電気的に再構成可能なゲートアレイロジックを用いる方法及び、これによって構成される装置

(57)【要約】
【課題】　電気的に再構成可能なゲートアレイを用いて論理構成を構築する方法を提供する。
【解決手段】　複数の電気的に再構成可能なゲートアレイ（ＥＲＣＧＡ）論理チップは、再構成可能な相互接続を介して相互に接続されている。そして、大規模デジタル回路網の電気的表現は、相互接続チップ上で一時的に実際に動作するハードウェア形態を採るように変換される。再構成接続により、相互接続チップ上に実現されたデジタル回路網は随時変更される。これによって、システムは、シミュレーション、プロトタイピング、実行、計算を含む種々の目的に適合される。再構成可能な相互接続は、相互接続機能専用のＥＲＣＧＡチップにより構成されている。各相互接続ＥＲＣＧＡは、複数の相互接続チップの全てではないが少なくとも一つのピンに接続されている。



【特許請求の範囲】

【請求項１】　第１及び第２の電気的に再構成可能なゲートアレイ（ＥＲＣＧＡ）を設ける工程と；ブールのロジックゲートから成る基本要素を具え、第１デジタルロジック回路網を表している第１入力データ及び、前記基本要素を相互接続する回路網を設ける工程と；前記第１入力データを、第１及び第２部分に分割する工程と；分割された第１データの第１部分を、第１ＥＲＣＧＡに供給し、これによって表現される前記第１ロジック回路網の第１部分が、第１ＥＲＣＧＡにおいて実際に動作形態を採るようにする工程と；前記分割された第１データの第２部分を、第２ＥＲＣＧＡに供給し、これによって表現される前記第１デジタルロジック回路網の第２部分が、第２ＥＲＣＧＡにおいて実際に動作形態を採るようにする工程と；前記第１及び第２ＥＲＣＧＡを相互接続し、前記第１入力データで特定される少なくとも一個の回路網が、前記第１及び第２ＥＲＣＧＡの間に及ぶようにする工程と；ブールのロジックゲートから成る基本要素と、前記基本要素を相互接続する回路網とを具えていることを除き、前記第１デジタルロジック回路網と全く無関係な、第２デジタルロジック回路網を表現している第２入力データを供給し、前記第１及び第２デジタルロジック回路網が同一のＥＲＣＧＡにおいて実際の動作形態を採るようにする工程と；前記第２入力データを、第１及び第２部分に分割する工程と；分割された第２データの第１部分を、第１ＥＲＣＧＡに供給し、これによって表現される前記第２デジタルロジック回路網の第１部分が、前記第１ＥＲＣＧＡにおいて実際に動作形態を採ることができるようにする工程と；前記分割された第２データの第２部分を、第２ＥＲＣＧＡに供給し、これによって表現される前記第２デジタルロジック回路網の第２部分が、前記第２ＥＲＣＧＡで実際に動作形態を採ることができるようにする工程と；前記第１及び第２ＥＲＣＧＡを相互接続し、前記第２入力データで特定される少なくとも１個の回路網が、前記第１及び第２ＥＲＣＧＡの間に及ぶように構成する工程；とを具えることを特徴とする方法。

【請求項２】　前記区分化の工程を、自動的に行うことを特徴とする請求項１に記載の方法。

【請求項３】　シミュレートされる第１デジタルロジック回路網を規定する工程と；前記第１デジタルロジック回路網を表現している第１入力データを発生させる工程と；前記第１入力データを、第１及び第２部分に分割する工程と；前記分割された第１データの前記第１部分を、前記第１ＥＲＣＧＡに供給し、このようにして表現される前記第１デジタルロジック回路網の第１部分が、前記第１ＥＲＣＧＡにおいて、実際に動作する形態を採ることができるようにする工程と；前記分割された第１データの前記第２部分を前記第２ＥＲＣＧＡに供給し、このようにして表現される前記第１ロジック回路網の第２部分が、前記第２ＥＲＣＧＡにおいて実際に動作する形態を採ることができるようにする工程と；前記第１及び第２ＥＲＣＧＡを相互接続し、前記第１入力データで特定される少なくとも１個の回路網が、前記第１及び第２ＥＲＣＧＡの間に及ぶようにする工程と；第１シミュレーションで使用する一組の第１刺激をソフトウェアで規定する工程と；前記刺激を規定するソフトウェアを、第１電気信号に変換する工程と；前記入力信号としての第１電気信号を、相互接続された前記第１及び第２ＥＲＣＧＡに供給する工程と；相互接続された前記第１及び第２ＥＲＣＧＡから、第１出力電気信号を受信する工程と；前記第１電気出力信号を、ソフトウェアの形態に変換する工程；とを具え、且つ、

前記第１デジタルロジック回路網とは別の第２デジタルロジック回路網において、上記工程を繰り返す工程；を具えている請求項１に記載のシミュレーション方法。

【請求項４】　合成ツールを使用して、第１コンピュータプログラムを、該第１コンピュータプログラムで表現されるアルゴリズムに従って動作する第１デジタルロジック回路網を表現している一組の第１入力データに変換する工程と；前記第１入力データを、前記第１及び第２部分に分割する工程と；前記分割された第１データの前記第１部分を、前記第１ＥＲＣＧＡに供給し、このようにして表現される前記第１デジタルロジック回路網の第１部分が、前記第１ＥＲＣＧＡにおて、実際の動作形態を採ることができるようにする工程と；前記分割された第１データの前記第２部分を前記第２ＥＲＣＧＡに供給し、このようにして表現される前記第１デジタルロジック回路網の第２部分が、前記第２ＥＲＣＧＡにおいて、実際の動作形態を採ることができるようにする工程と；前記第１及び第２ＥＲＣＧＡを相互接続し、前記第１入力データが特定する少なくとも一個の回路網が、前記第１及び第２ＥＲＣＧＡの間に及ぶようにする工程と；前記第１プログラムの入力データに対応する第１刺激信号を発生させる工程と；前記第１刺激信号を、入力信号として、相互接続された前記第１及び第２ＥＲＣＧＡに供給する工程と；相互接続された、前記第１及び第２ＥＲＣＧＡから、前記第１プログラムの出力データに対応する第１出力電気信号を受信する工程；とを具え、且つ、前記第１デジタル回路網とは別の第２デジタル回路網において、上記工程を繰り返す工程；を具えている請求項１に記載の計算方法。

【請求項５】　前記合成ツール使用工程が：設計部合成ツールを使用して、前記第１コンピュータプログラムを、データパス及び有限状態マシンコントローラから成り、前記第１プログラムによって表現されるアルゴリズムに従って作動するシステムの表現に変換する工程と；ロジック合成ツールを用いて、前記設計部合成ツールによって提供されるデータパス及び有限状態マシンコントローラの表現を一組の第１入力データに変換する工程；

とを具えることを特徴とする請求項１に記載の方法。

【請求項６】　前記ＥＲＣＧＡが、各々、複数のピンを具え、且つ、前記相互接続の工程が：少なくとも一個の追加のＥＲＣＧＡを設け、再構成可能な相互接続としての役割を果たすようにする工程と；前記再構成可能な相互接続ＥＲＣＧＡの各々を、前記第１及び第２ＥＲＣＧＡのピンのすべてではないが、少なくとも一個に接続する工程；とを具えていることを特徴とする請求項１に記載の方法。

【請求項７】　(ａ)Ｎ個のＥＲＣＧＡを設ける工程と；(ｂ) 前記第１入力データを、Ｎ個の部分に分割する工程と；

(ｃ)分割されたデータの各部分を、対応するＥＲＣＧＡに供給し、このようにして表現される前記デジタルロジック回路網の前記部分が、前記ＥＲＣＧＡにおいて実際の動作形態を採ることができるようにしている工程と；

(ｄ)Ｎ個のＥＲＣＧＡを相互接続し、ＥＲＣＧＡの各々を、少なくとも一個の他のＥＲＣＧＡに接続し、且つ、前記入力データで特定される回路網の各々を実現する工程と；

(ｅ)前記第２入力データに対して、(ｂ)、(ｃ)及び(ｄ)の工程を繰り返す工程；とを更に具えていることを特徴とする請求項１に記載の方法。

【請求項８】　ＥＲＣＧＡが各々複数のピンを具え、且つ、前記相互接続の工程が：少なくとも一個の追加のＥＲＣＧＡを設け、再構成可能な相互接続として役割を果たすようにする工程と；前記再構成可能な相互接続ＥＲＣＧＡの各々を、前記複数であるＮ個のＥＲＣＧＡのピンのすべてではないが、少なくとも個に接続する工程；とを具えていることを特徴とする請求項７に記載の方法。

【請求項９】　前記再構成可能な相互接続ＥＲＣＧＡを、前記Ｎ個の各々のＥＲＣＧＡの全てではないが、少なくとも一個に接続する工程を、更に具えていることを特徴とするむ請求項８に記載の方法。

【請求項１０】　順次、というよりはむしろ単一のプロセスで：前記入力データを分割する工程と；これによって、対応して必要となる相互接続の特徴を識別する工程；とを実行する工程を更に具え、これによって、前記入力データを、このような方法で分割し、対応して必要な相互接続を簡易化することを特徴とする請求項９に記載の方法。

【請求項１１】　シード基本要素を決定し、他の基本要素をこれに加えることによって前記入力データを分割し、これによって、基本要素のクラスタを構成する工程を更に具え；前記基本要素の各々が、多数のピンを具え；前記クラスタの構成が、クラスタに割り当てられていない各基本要素の有効な機能を評価する工程を具え；前記有効な機能が、最も多数のピンを有する基本要素に、最大の初期利益をもたらすことを特徴とする請求項１０に記載の方法。

【請求項１２】　シード基本要素を決定するとともに、他の基本要素をこっちに加えることによって、前記入力データを分割し、これによって、基本要素のクラスタを構成する工程と；相互接続の限界に至るまで、基本要素をクラスタから取り除く工程とを更に具え；前記分割の工程が、相互接続の限界を超えて、基本要素をクラスタに加える方法を備えていることを特徴とする請求項１０に記載の方法。

【請求項１３】　前記相互接続の工程が、更に：２個のＥＲＣＧＡ間に及んでいる回路網が：回路網のルートを決定することのできる複数の対象となる再構成可能な相互接続ＥＲＣＧＡを試験する工程と；少なくとも部分的に、ＥＲＣＧＡがすでに用いられている利用の程度に基づき、このような相互接続ＥＲＣＧＡの各々を介してのルート決定の適性を評価する工程；とを具えていることを特徴とする請求項１０に記載の方法。

【請求項１４】　(ａ)Ｎ個のＥＲＣＧＡを設ける工程と；

(ｂ)回路形態的に、前記Ｎ個のＥＲＣＧＡを、規則的な多次元アレイに配置し、これによって相対的に隣接するＥＲＣＧＡを決定する工程と；

(ｃ)直接的に隣接するＥＲＣＧＡを相互接続する工程と；

(ｄ)前記第１入力データを、Ｎ個の部分に分割する工程と；

(ｅ)前記分割されたデータの各部分を、対応するＥＲＣＧＡに供給し、このようにして表現される前記デジタルロジック回路網の前記部分が、前記ＥＲＣＧＡにおいて実際に動作する形態を採ることができるようにする工程と；

(ｆ)非隣接ＥＲＣＧＡ間に介在するＥＲＣＧＡを介して相互接続を確立することで、非隣接ＥＲＣＧＡを相互接続し、必要されるＮ個のＥＲＣＧＡを相互接続し、前記第１データで特定される回路網を実現する工程と；

(ｇ)前記第２入力データに対して、工程(ｄ)、(ｅ)及び(ｆ) を繰り返す工程；とを更に具えていることを特徴とする請求項１に記載の方法。

【請求項１５】　自動ルーティング方法を用いて、非隣接ＥＲＣＧＡを相互接続するのに、介在するどのＥＲＣＧＡ及びピンを用いるかを決定する工程を更に具えていることを特徴とする請求項１４に記載の方法。

【請求項１６】　前記デジタルロジック回路網中の故障を、前記入力データによって表現することで、故障をシミュレートする工程を更に具えていることを特徴とする請求項１に記載のフォールトシミュレータに関する方法。

【請求項１７】　電気設計自動化システムと接続している相互接続されたＥＲＣＧＡを作動させる工程を更に具えていることを特徴としている請求項１に記載の方法。

【請求項１８】　前記相互接続されたＥＲＣＧＡを、メモリ回路に結合させる工程と、前記回路と接続している前記相互接続されたＥＲＣＧＡを動作させる工程とを更に具える請求項１に記載の方法。

【請求項１９】　前記双方向回路網を、双方向性相互接続を用いて、積の和に変換することによって、双方向性回路網を相互接続する工程を更に具えることを特徴とする請求項１に記載の方法。

【請求項２０】　ＥＲＣＧＡにおいて積を加算する工程を更に具えていることを特徴とする請求項１に記載の方法。

【発明の詳細な説明】

【０００１】

【発明の分野】本発明は、電気的に再構成可能なゲートアレイロジック素子（ＥＲＣＧＡ）の使用に関するとともに、複数のこのようなロジック素子の相互接続を具え、ラージデジタル回路網の電気的な表現を、シミュレーション、プロトタイピング、実行及び／又は演算のための相互接続されたロジック素子を用い、一時的に実際動作するハードウエアの構成に変換する方法にも関するものである。

【０００２】

【発明の背景及び概要】説明の便宜上、本出願では、リアライザシステム（Realizer System）として本発明を説明する。辞書には、後述するシステムの簡単な説明的名称が欠けている。リアライザシステムは、ハードウエアとソフトウエアとを具えており、シミュレーション、プロトタイピング、実行又は演算のために、ラージデジタルロジック回路網の表現を、一時的に実際動作するハードウエアの構成に変換する。（数個の最も広く利用することのできる構成可能なロジックデバイスを用いることによって、極めて多くのロジック機能を具えている場合、デジタルロジック回路網をラージとみなすこととしている。）（専らではなく）一般的に用いられている適切な用語を簡単に再検討することで、以下の説明を、より理解し易いものとすることができる。何かを“実現する”とは、それを実際又は現実のものとすることである。デジタルロジック回路網又は設計の全体又は一部を実現するということは、それを永久的に組み立てることなく、実際の動作を構成することである。“入力設計部”とは、実現されるべきデジタルロジック回路網を表している。この入力設計部は、計測デバイス又はユーザ指定実デバイスと同様に、組合せロジック及び記憶を表している基本要素と、基本要素の入出力ピン間の接続を表現している回路網とを具えている。ロジックチップ又は相互接続チップを“構成”するとは、その内部ロジック機能及び／又は相互接続を特定の方法で配置することである。入力設計部のためのリアライザシステムを構成するとは、その内部ロジック機能及び相互接続を、入力設計部に応じて配置することをいう。設計を“変換”するとは、その表現を構成データのファイルに変換することであり、これをリアライザハードウエアに直接用いると、設計部を実現することができる。設計部を“作動”させるとは、入力設計部の表現に応じて構成されたリアライザハードウエアを実際に作動させることである。“相互接続”とは、ピンがワイヤで相互接続されているかのように、多数のチップＩ／Ｏピン間にロジック信号を通過させるための再構成可能な手段である。“バス”とは、部分的にクロスバー相互接続におけるロジックチップとクロスバーチップとの間の又は部分的クロスバーの階層におけるクロスバーチップ間の組込相互接続ワイヤの中の一つをいう。“バスナンバ”は、一対のチップを相互に接続している多くのバスの中から特定のバスを特定するものである。“ＥＲＣＧＡ”とは、電気的に再構成可能なゲートアレイ、すなわち組合せロジックの捕捉及び入力／出力接続（及び付加的な記憶装置）のことであり、その機能及び相互接続は、単に電気信号を供給することにより何回にも亘って構成及び再構成される。“ロジックチップ”とは、リアライザシステムにおける入力設計部の組合せロジックと、記憶装置と、相互接続とを実現するのに用いられるＥＲＣＧＡである。“Ｌチップ”とは、ロジックチップ、又はロジックチップの場所に取り付けられるメモリモジュール、又はユーザ指定のデバイスモジュールである。“相互接続チップ”とは、Ｉ／Ｏピン間の任意の相互接続を実行することのできる電気的に再構成可能なデバイスである。“ルーティングチップ”とは、直接相互接続、又はチャンネルルーティング相互接続に用いられる相互接続チップのことである。“クロスバーチップ”とは、クロスバー相互接続又は部分的クロスバー相互接続に用いられる相互接続チップである。“Ｘチップ”とは、Ｌチップを相互に接続する部分的クロスバー中のクロスバーチップである。“Ｙチップ”とは、階層部分的クロスバー相互接続の第２レベルにおけるクロスバーチップであり、Ｘチップを相互に接続している。“Ｚチップ”とは、階層部分的クロスバー相互接続の第３レベルにおけるクロスバーチップであり、Ｙチップを相互に接続している。“ロジックボード”とは、ロジックを伝達するプリント回路ボード及び相互接続チップである。“ボックス”とは、１以上のロジックボードを具えているカードケージのような物理的格納装置である。“ラック”とは、１以上のボックスを具えている物理的格納装置である。“システムレベル相互接続”とは、個々のチップより大きなデバイスを相互接続することであり、ロジックボード、ボックス、ラック等である。“ロジックセルアレイ”又は“ＬＣＡ”とは、ＥＲＣＧＡの特定の例であり、Xilink Inc.その他で製造され、好適な例に用いられるものである。“構成可能なロジックブロック”又は“ＣＬＢ”は、構成可能なロジックの小さなブロック及びフリップフロップであり、ＬＣＡにおける組合せロジック及び記憶を表

している。“設計メモリ”とは、入力設計部において特定されるメモリ機能を実現するメモリデバイスである。“ベクトルメモリ”とは、多くの刺激信号をリアライザシステムに供給及び／又はリアライザシステムにおいて実現される設計部からの多くの応答信号を捕捉するのに用いられるメモリデバイスである。“スティミュレータ（stimulator）”とは、刺激信号を実現された設計部の個々の入力端子に供給するのに用いられるリアライザシステム中のデバイスである。“サンプラ”とは、実現された設計部の個々の出力端子からの応答信号を捕捉するのに用いられるリアライザシステム中のデバイスである。“ホストコンピュータ”とは、リアライザシステムのホストインターフェイスハードウエアが接続されている慣用のコンピュータシステムであって、リアライザのハードウエアの構成及び動作を制御するものである。“ＥＤＡシステム”とは、電気自動設計システム、すなわち電気設計部を作成、編集、及び分析するのに用いられるコンピュータベースのツールに関するシステムである。ホストＥＤＡシステムとは、リアライザシステムを応用する多くの場合において、入力設計ファイルを発生させるものである。

【０００３】シングルラージ設計部を保持するのに十分な容量を有する再構成可能なゲートアレイを用いれば、リアライザ技術の多くは不要である。しかしながら、２つの理由からこのことは決して実現することができない。まず第１に、ＥＲＣＧＡのロジック容量は、同じ製造技術を用いて製造された物理的に同一のサイズの再構成不可能な集積回路と同じではない。再構成のための機能は、チップのかなりのスペースをとる。ＥＲＣＧＡは、信号を導くスイッチングトランジスタと、これらのスイッチを制御するための記憶トランジスタとを有しており、ここで、再構成不可能なチップは金属トレースを具えている。そして、ＥＲＣＧＡは、これらのトランジスタをロジックとして用いることができる。再構成可能なチップに必要とされる規則性は、リソースが実際の設計では用いられていないということである。その理由は、規則的なロジック構造の配置及びルーティングが、利用可能なゲートを１００％用いることができないからである。ＥＲＣＧＡを製作するためのこれらの計数の結合は、再構成することのできないチップのロジック容量の約１／１０である。目下のところ、実際の実行では、ＥＲＣＧＡに要求される最大ゲート容量は９，０００ゲートである（Xilinx XC 3090）。同様の技術を用いて製造された、現在ほぼ慣用となっている集積回路では、１００，０００ゲートロジック容量以上が与えられる（モトローラ）。第２に、通常１０～１００或いはそれ以上の多くの集積回路を用いて、多くのプリント回路ボード上にリアルデジタルシステムを設けることはよく知られていることである。ＥＲＣＧＡのロジック容量が、大規模集積回路と同等である場合であっても、ほとんどのデジタルシステムを実現するのに、依然としてこのようなチップを多く用いることとなる。しかし、ＥＲＣＧＡのロジック容量は、大規模集積回路と同等ではないため、更に多くのチップが必要となる。最終的に、リアライザシステムがシングル大規模チップのロジック容量を有するためには、リアライザシステムが１０のオーダのＥＲＣＧＡを有している必要がある。このリアライザシステムが、このようなチップの容量を有しているためには、１００個のオーダのＥＲＣＧＡが必要とされる。このことは、特殊な製造技術とは無関係に必要とされることに注意しなければならない。チップ当たりのトランジスタの数を２倍にすることによる製造工程によって、ＥＲＣＧＡの能力を２倍にすると、再構成不可能なチップの容量は２倍となり、従って、設計サイズ全体も同様に２倍となる。

【０００４】これらの理由によって、有効なリアライザシステムを開発するためには、電気的に再構成可能な方法で、数百個のＥＲＣＧＡを相互接続できるようにするとともに、設計を数百個のＥＲＣＧＡの構成に変換できるようにする必要がある。本発明は、ＥＲＣＧＡそれ自体の技術にまで及ぶものではなく、多くのＥＲＣＧＡからリアライザシステムを開発するための技術にのみ関するものである。ＥＲＣＧＡ技術は、いかにしてリアライザシステムを開発するかを示してはいない。その理由は、それが別の問題だからである。一つのＩＣチップの全体を構成している再構成可能な相互接続ロジック素子のためのＥＲＣＧＡ技術は、多くを相互接続するのに適用できない。いずれか一方の方向に信号を通すスイッチングトランジスタによって、ＥＲＣＧＡ相互接続を容易に達成することができる。一つのチップ全体に亘って障壁が存在しないので、相互接続に利用する多数の通路が存在する。チップが小さいので信号のディレイも小さい。多くのＥＲＣＧＡを相互接続するのはむずかしい。その理由は、ＩＣパッケージピン及びプリント回路ボードを伴っているからである。利用することのできるピンの数が制限されているということは、相互接続のための通路の数が制限されているということである。チップの信号の入出力は、（例えば増幅しながら）アクティブピンバッファを介して行われなければならない。アクティブピンバッファは、一方向にのみ信号を送ることができる。これらのバッファ及び回路基板はトレースによって、一つのチップによって生ずるディレイよりも大きなディレイが生ずるリアライザシステムの相互接続技術によって、ＥＲＣＧＡとは全く別の方法で、これらの問題を解決する。最終的に、ＥＲＣＧＡ技術では、設計を多くのチップの構成に変換する必要はない。リアライザシステムの相互接続は、ＥＲＣＧＡ内の相互接続とは全く異なるものであり、相互接続を決定及び構成する全く別の方法が必要とされる。所定の時間に利用することのできる迅速且つ緻密なシリコン技術を用いて、ＥＲＣＧＡ

を開発する。（１ミクロンＳＲＡＭ技術を用いて、1989 Xilinx XC3000 ＬＣＡを開発する。）これは、実現される迅速且つ緻密なシステムと同一の技術である。ＥＲＣＧＡは汎用のものであり、再構成可能な相互接続を具えているので、現行のゲートアレイや慣用のチップほど緻密ではないファクタを具えている。リアライザシステムは、ＥＲＣＧＡのレベルより高い汎用性及び再構成可能性に対するサポートを反復する。従って、リアライザシステムは、現行の緻密なシステム程緻密ではなく、常に概略的には、１のオーダの一定のファクタとなっている。ボードレベルのリアライザシステムは、ゲートアレイを実現し、ボックスレベルのリアライザシステムは、ボード及び大規模慣用チップを実現する。更に、ラックレベルのリアライザシステムは、ボックスを実現する。設計構造は、パッケージングの影響を強く受ける。Ｉ／Ｏピン幅：ＶＬＳＩチップレベルでは、１００個のＩ／Ｏピンは容易に開発でき、２００ピンは開発が困難であるが、用いないこともなく、４００ピンに関しては、ほとんど開発されていない。ボードレベルでは、これらの数字は概して２倍となっている。ロジック密度：ボードが、しばしば５個のＶＬＳＩチップを具えており、１０個のＶＬＳＩを具えることも可能であるも、２０個のＶＬＳＩを具えることは一般的ではない。単にその理由は、実際のボードの最大値が約２００平方インチに制限されているからである。ボックスは、通常１０～２０ボードを具えており、時には４０ボードを具えている。相互接続密度：２次元ワイヤの平面を数枚用いることができる場合、モジュールは、完全にチップ及びボード上で相互接続される。しかし、背面が本質的に一次元的な場合には、ボックスレベルにおいて、それほど完全に相互接続されているわけではない。これらパッケージングの制約は、有用なリアライザシステムにおいて見られるシステム構造にかなり影響を及ぼす。リアライザシステムでは低密度であるがために、実現される設計部では、単一のロジックチップは、通常、唯一のモジュールを構成する。一つのボードにおけるロジックチップの複合体によって、１つ又は２つのＶＬＳＩチップを実現する。リアライザボードのボックスは、設計部において、単一のボードを実現する。更に、ボックスのラックは、設計部のボードのボックスを実現する。従って、リアライザシステムのボードレベルのロジック及び相互接続の複合体は、設計部のＶＬＳＩチップと同じロジック・相互接続容量並びにＩ／Ｏピン幅を有している必要がある。リアライザシステムのボックスは、設計部のボードと同じロジック・相互接続容量及びＩ／Ｏピン幅を必要とし、リアライザシステムのラックは、設計部のボックスと同じロジック・相互接続容量を必要としている。

【０００５】

【発明の実施の形態】内容一覧表

【０００６】1．リアライザハードウエアシステム

リアライザハードウエアシステム（図1）は：

１）１）少なくとも二つのロジックチップ（通常、数十個又は数百個）及び

２）付加的に、メモリモジュール、ユーザ指定のデバイスモジュールのような、１以上の特定目的のための構成要素を具えている１セットのＬチップを、

２）Ｉ／Ｏピンを相互接続可能なすべてのＬチップに接続されている構成可能な相互接続と、

３）ホストコンピュータ、構成システム及びデータ入出

力又は制御のためのホストが使用することのできるすべてのデバイスに接続されたホストインタフェースと、
4）ホストインタフェース、すべての構成可能なLチップ及び相互接続デバイスに接続された構成システムとを具えている。このハードウエアを、通常、ロジックボード、ボックス及びラックの形態で実装し、ホストコンピュータに接続する。このハードウエアは、ホストコンピュータの制御の下で作動する。

【０００７】1.1　ロジック及び相互接続チップ技術

1.1.1　ロジックチップデバイス

デバイスが、リアライザロジックチップとして役立つためには、このデバイスが電気的に再構成可能なゲートアレイ（ＥＲＣＧＡ）でなければならない：
1）デバイスは、容量制限を条件として、組合せロジック（及び付加的な記憶装置）を具えているデジタルロジック回路によって構成することができなければならない。
2）デバイスは、その機能及び内部相互接続を、電気的に何回でも、種々の論理回路に適合するように構成することができるという点において、電気的に再構成可能でなければならない。
3）デバイスは、特定の回路網又は特定するＩ／Ｏピンとは無関係に、デジタル回路網でＩ／Ｏピンを自由に接続し、リアライザシステムの部分クロスバー、又は直接相互接続が首尾よくロジックチップを相互接続できなければならない。ロジックチップとして好適な、再構成可能なロジックチップの一例としては、ロジックセルアレイ（Logic Cell Array（ＬＣＡ））がある（“The Programmable Gate Array Handbook”,Xilinx Inc.,San Jose,CA,1989)。このロジックチップは、Xilinx Inc.その他で製造されている。このチップは、構成可能なロジックブロック（Cnofigurable Logic Block（ＣＬＢ））の２次元配列を具えている。この２次元配列は、再構成可能なＩ／Ｏブロック（ＩＯＢ）で囲まれているとともに、ＣＬＢとＩＯＢとの間の行と列とに配置されたセグメントを配線することによって相互接続されている。各ＣＬＢは、数個の入力端子と、ロジック機能を再構成することのできる多重入力組合せロジック回路網と、１以上のフリップフロップと、ＣＬＢ内の再構成可能な相互接続によって連結することができる１以上の出力端子とを具えている。各ＩＯＢを再構成し、チップの入力バッファ又は出力バッファとすることができる。更に、各ＩＯＢを外部Ｉ／Ｏピンに接続する。配線したセグメントをＣＬＢ、ＩＯＢ及び、お互いに接続することによって、再構成可能なパストランジスタ及び相互接続マトリックスを介し、ＣＬＢ、ＩＯＢ及びセグメント間に、相互接続を形成する。すべての再構成可能な機能を、チップのシリアルシフトレジスタにおけるビットで制御する。従って、ＬＣＡは“構成ビットパターン”のシフトによって完全に構成される。構成に要する時間は１０～１００マイクロ秒である。Xilinx 2000及び3000シリーズのＬＣＡは、６４～３２０のＣＬＢを具えており、５６～１４４のＩＯＢを使用することができる。ＬＣＡネットリスト（netlist）変換ツール（以下にて示す）は、ロジックをＣＬＢに作成し、ＣＬＢとＩＯＢとの間の相互接続を最適にしている。ＣＬＢとＩ／Ｏピンとの間の相互接続を構成することによって、ＬＣＡは、特定の回路網又は特定するＩ／Ｏピンとは無関係に、デジタル回路網でＩ／Ｏピンを自由に接続することができる。リアライザシステムを好適に具体化するには、ＬＣＡデバイスをそのロジックチップとして用いる。ロジックチップとして好適な他の種類のアレイとしては、ＥＲＡ、すなわち電気的に再構成可能なアレイがある。市販されているものとしてはplesseyのERA60Kのタイプのデバイスがある。これは、構成ビットパターンを部分的にＲＡＭにロードすることで構成される。ＥＲＡを２入力ＮＡＮＤゲートのアレイとして構成する。ＲＡＭの値に応じて、２入力ＮＡＮＤゲートの各々を独立に互いに相互接続する。ＥＲＡは、ゲート入力端子の一連の相互接続通路への接続を切換える。ERA 60100は、約１０，０００個のＮＡＮＤゲートを具えている。アレイの周辺のＩ／Ｏセルは、ゲート入力端子及び／又は出力端子を外部Ｉ／Ｏピンに接続するのに用いられる。ＥＲＡネットリスト変換ツールは、ロジックをゲートに作成し、ゲート間の相互接続を最適にするとともに、以下に示すように、構成ビットパターンファイルを出力する。ゲートとＩ／Ｏセル間の相互接続を構成可能にすることによって、ＥＲＡは、特定の回路網又は特定するＩ／Ｏピンとは無関係にデジタル回路網を用いてＩ／Ｏピンを自由に接続することができる。ロジックチップとして用いることのできる、更に他の種類の再構成可能なロジックチップとしては、ＥＥＰＬＤ、すなわち、電気的に消去可能で、プログラム可能なロジックデバイス（“GAL Handbook”; Lattice Semiconductor Corp., Portland, OR,1986）がある。商用のものとしては、ラティス・ジェネリック・アレイ・ロジック（Lattice Generic Array Logic（ＧＡＬ））がある。これは、ビットパターンを、ロジック構成の部分にロードすることで構成される。ＧＡＬは、出力フリップフロップを有する、積の和のアレイとして構成されており、その構成は、Xilinx ＬＣＡよりも汎用性を有していない。ＧＡＬによって、Ｉ／Ｏピンを、すべての入力ピン間及びすべての出力ピン間のロジックに接続せずにすみ、部分的に要件を満足している。ＧＡＬは、１０～２０個のピンを有しており、比較的小さな構造となっている。しかし、ＧＡＬはリアライザロジックチップとして用いられる。プログラム可能なロジックチップについての詳細は、米国特許第４，６４２，４８７号、第４，７００，１８７号、第４，７９６，２１６号、第４，７２２，０８４号、第４，７２４，３０７号、第４，７５８，９８５号、第４，７６８，１９６

号及び第４，７８６，９０４号明細書において説明されている。ここでは、これらの明細書の内容を説明に用いている。

【０００８】１．１．２　相互接続チップデバイス

相互接続チップは、クロスバー相互接続の全体及び一部に用いるクロスバーチップと、直接相互接続及びチャンネルルーティング相互接続に用いるルーティングチップとを具えている。デバイスが、リアライザ相互接続チップとして役立つためには：

１）デバイスは、直ちに、任意に選択されたＩ／Ｏピンのグループ間で多くのロジック相互接続を形成し、各相互接続は、その入力Ｉ／Ｏピンからロジック信号を受信するとともに、これらの信号を出力Ｉ／Ｏピンに供給できなければならない。

２）デバイスは、その相互接続を電気的に規定するという点において、再構成可能でなければならず、多くの種々の設計に適合できるように再規定できなければならない。

３）クロスバー加算技術を用いて、部分的クロスバー相互接続におけるトライステート回路網を相互接続する場合、デバイスは、加算ゲートを具体化できなければならない。（クロスバー加算技術を用いない場合には、トライステート・セクションにて説明するように、他のトライステート技術を用いる。）

上述したＥＲＣＧＡデバイス、すなわち、ＬＣＡ、ＥＲＡ及びＥＥＰＬＤは、これらの要件を満足しており、相互接続チップとして用いられる。相互接続チップにロジックをほとんど或いは全く用いない場合、ほとんどのデジタル回路網を構成することのできる機能は、データを直接入力ピンから出力ピンへと送ることができる。ＬＣＡは、リアライザシステムを好適に具体化する際、クロスバーチップとして用いられる。TI　74AS　8840デジタルクロスバースイッチ（SN 74 AS 8840 Data Sheet, Texas Instruments, Dallas, TX, 1987）、すなわち、通常電話スイッチに用いられる交差点スイッチデバイスを、相互接続チップとして用いることができる。ところで、これらのクロスバースイッチデバイスを、作動中、動的に変化する構成に応用する場合、データ伝送スピードに匹敵する再構成スピードが得られる。この再構成スピードは、ＥＲＣＧＡデバイスの構成スピードよりも速い。この結果、このようなクロスバースイッチデバイスは、ＥＲＣＧＡよりも高価且つ低容量であり、あまり望ましくないリアライザ相互接続チップを作成することとなってしまう。

【０００９】１．１．３　ＥＲＣＧＡ構成ソフトウエア

構成ビットパターンは、ユーザ指定に従ってＥＲＣＧＡにロードされ、そのロジックを構成する。ユーザが単独でロジックを構成するのは非現実的である。従って通常ＥＲＣＧＡ装置を製造することによって、ネットリスト変換ソフトウエアツールが得られる。このツールは、ネットリストファイルに具わっているロジック仕様を、構成ビットパターンファイルに変換する。リアライザ設計変換システムは、ＥＲＣＧＡのコンピュータメーカによって提供されるネットリスト変換ツールを用いている。リアライザ設計変換システムが、設計部において、ネットリスト変換ツールを読出して変換し、ロジックチップに分割し、さらに相互接続を決定すると、各ロジックに対するネットリスト及びリアライザハードウエアにおける相互接続チップを発生させる。ネットリストファイルとは、すべての基本要素（ゲート・フリップフロップ及びＩ／Ｏバッファ）及び、単一ロジックチップ又は相互接続チップで構成されるこれらの相互接続のリストのことである。リアライザ設計変換システムは、ＥＲＣＧＡネットリスト変換ツールを各ネットリストファイルに供給し、各チップの構成ファイルを得る。ロジックチップ及び相互接続チップとして種々のデバイスを用いる場合には、適切なツールを用いる。構成ファイルは、２進ビットパターンを具え、これはＥＲＣＧＡデバイスにロードされると、ネットリストファイルの仕様に応じてファイルを構成する。ＥＲＣＧＡデバイスは、これらのファイルを、永久に記憶され且つ作動前に設計部のリアライザシステムを構成するのに用いられる単一バイナリーファイルに収集する。リアライザ設計変換システムは、ツールのＥＲＣＧＡコンピュータメーカによって規定されるネットリスト及び構成ファイルフォーマットに準拠している。

【００１０】１．１．４　ネットリスト変換ツール

リアライザシステムを好適に具現化するために、ロジックチップ及びクロスバーチップとしてＬＣＡを用いているので、Xilinx　ＬＣＡネットリスト変換ツール及びそのファイルフォーマットをここで説明する。Xilinx製のＬＣＡネットリスト変換ツール（ＸＡＣＴ）によって、ネットリスト形式のロジック回路網が与えられるとともに、自動的にロジック素子がＣＬＢに作成される。Ｉ／Ｏピンの位置に関し、最適の方法でロジック素子を構成し、内部相互接続を容易にすることができる。従って、このツールは、いかにしてロジックチップの内部相互接続を構成するかを解明し、その出力結果としての構成ファイルを作成する。ＬＣＡネットリスト変換ツールは、単に個々のＬＣＡを変換するだけであって、ロジック回路網が大き過ぎて単一のＬＣＡに適合できない場合には障害が生じる。Xilinx　ＬＣＡネットリストファイルをＸＮＦファイルと称する。これはアスキーテキストファイルであり、各々の基本要素に対する１セットのＸＮＦファイル中のステートメントを具え、基本要素、ピン及びこれらのピンに接続される回路網の名称を特定する。これらの回路網は、ＬＣＡネットリスト中で相互接続されており、入力設計部の回路網ではなく、ＬＣＡ基本要素を接続している。ＸＮＦファイル中のいくつかのファイルは、設計変換の結果、入力設計部の回路網に直接対

応しているが、他のファイルは対応していない。例えば、これらは‘Ｉ＿１７８１‘と称する、２入力ＸＯＲゲートを特定するためのＸＮＦファイル基本要素ステートメントであり、前記２入力ＸＯＲゲートの入力ピンを、‘ＤＡＴＡ０’及び‘ＩＮＶＥＲＴ’と称する回路網に接続し、その出力ピンを、‘ＲＥＳＵＬＴ’と称する回路網に接続している：

```
SYM,I_1781,XOR
PIN,O,O,RESULT
PIN,1,I,DATA0
PIN,0,I,INVERT
END
```

入力及び出力Ｉ／Ｏピンバッファ（入力のためのＩＢＵＦ及び出力のためのＯＢＵＦ）は、Ｉ／Ｏピンを特定するためのステートメントを付加することで、同様にして特定される。これらは、ＯＢＵＦに対する基本的なステートメントであり、これによって、‘ＲＥＳＵＬＴ’を‘ＲＥＳＵＬＴ＿Ｄ’と称する回路網を介してＩ／Ｏピン‘Ｐ５７’において駆動させる：

```
SYM,IA_1266,OBUF
PIN,O,O,RESULT_D
PIN,I,I,RESULT
END
EXT,RESULT_D,O,LOC=P57
```

Xilinx　ＬＣＡを、ＲＢＴファイルと称する。これは、アスキーテキストファイルであり、構成される部分を識別するヘッダステートメントと、動作のための部分を構成するのに用いられるバイナリービットパターンを特定する‘０’及び‘１’のストリームとを具えている。

【００１１】1.2　相互接続アーキテクチャ

実際の場合、大規模入力設計部を実現するためには、多くのロジックチップを使用しなければならないため、リアライザのロジックチップを再構成可能な相互接続に接続しなければならない。この相互接続によって、必要とされているように、設計部中の信号が、分離ロジックチップ間を流れる。この相互接続は、電気的相互接続及び／又は相互接続チップの結合を具えている。リアライザシステムにおいて大規模設計部を実現するためには、トータルで幾万ものＩ／Ｏピンを有するロジックチップが相互接続によって供給されなければならない。相互接続は、システムサイズが大きくなるにつれて経済的に拡張可能であり且つ容易なものであるとともに、入力設計部を幅広く確実に構成することができ、更に高速であり、ロジックチップ間の遅延を最小にすることができる。現実の設計部における回路網単位のピンの平均数は、設計部のサイズとは無関係な小さな数であるため、接続するロジックチップのトータル数が増加するにつれて、優れた相互接続のサイズ及びコストも直接的に増加するはずである。設計部の容量が増大するにつれて、用いる特定ロジックチップ容量、ロジックチップの数及びロジックチップピンの数も直接的に増大する。従って、設計部の容量とともに、優れた相互接続のサイズ及びコストも直接的に変化する。２クラスの相互接続構造を説明する：隣接相互接続を第１セクションで説明し、クロスバー相互接続を次のセクションで説明する。最も近い隣接相互接続は、ロジックチップと、２次元、３次元又はそれ以上の次元の面に従って、混合及び構成された相互接続とによって構成される。最も近い隣接相互接続では、ゲートアレイチップの行列編成又はプリント回路基板をロジックチップの編成にまで拡張している。所定の入力設計部の構成は、チップ及びボードを開発する場合に用いられるのと同様の配置及びルーティングプロセスによって決定される。クロスバー相互接続は、相互接続されているロジックチップとは異なる。クロスバー相互接続は、伝送及び演算に用いられるクロスバーの多入力多出力編成に基づくものであり、平面的に構成することができる。最も近い隣接相互接続は、ロジック容量が大きくなるにつれて大きくなるが、ルーティング通路が密集するにつれて大規模相互接続はゆっくりとなり、また、構成を決定することが困難且つ不確実なものとなる。単なるクロスバーは、その直接性のために極めて高速であり、その規則性のために構成が容易であるが、すぐに非実用的な大きさとなってしまう。部分的なクロスバー相互接続は、ほとんどの直接性と、単なるクロスバーの規則性とを保持するが、設計部容量の増大とともにのみ直接的に増大し、理想リアライザ相互接続を実現している。実際のリアライザシステムでは、図示した以外の相互接続を用いることはできるが、部分的クロスバーを好適な具体例に用いる。この使用は、この明細書全体を通して類推される。

【００１２】1.2.1　最も近い隣接相互接続

1.2.1.1　直接相互接続

直接相互接続では、相互接続チップを用いずして、直接すべてのロジックチップを、規則的なアレイにおいて相互に接続する。この相互接続は、単にロジックチップ間の電気的接続から成っている。ロジックチップの相互接続は、多くの異なるパターンを形成することができる。一般的に、一つのロジックチップのピンをグループ毎に分割する。従って、すべてのロジックチップにおいて各ピンのグループを、他のロジックチップの同様なピンのグループ等に接続する。各ロジックチップは、単に一組のすべてのロジックチップ、すなわち、物理的な意味において、つまり、少なくともアレイの接続形態という意味において、最も近い隣接ロジックチップとだけ接続する。１以上のロジックチップにロジックを接続するすべての入力設計部回路網を、相互接続チップとしての機能を果たす他のロジックチップに、これらのロジックチップを直接接続する場合には直接接続し、又は、一連の他のロジックチップを介して接続し、チップの実現するロジックのいずれかに接続することなしに、ロジック信号

を一方のＩ／Ｏピンから他方のＩ／Ｏピンへ伝達する。このようにして、いかなる所定のロジックチップも、設計部ロジックの共有に加えて、一方のチップから他方のチップへ相互接続信号を伝達するように構成されている。相互接続機能を実行することのできない非ロジックチップリソースを、アレイの周辺で、専用ロジックチップピンに接続又は、ロジックピンを相互に接続しているピンに正接させ接続している。図２に示す特定の例では、行及び列の２次元格子中に配置されたロジックチップを具えており、その各々のチップは、メモリを有する隣接ロジックチップに北側、南側、東側及び西側で接続されている４つのピンのグループと、Ｉ／Ｏと、周辺で接続されたユーザ指定のデバイスとを具えている。この相互接続を、ここで説明した２次元のものから、より高次元のものへと拡張することができる。一般的に、‘ｎ’を次元の数とする場合、各々のロジックチップのピンを、２＊ｎ個のグループに分割する。各々のロジックチップは、規則的な形態で、２＊ｎ個の他のロジックチップと接続している。他の変更も同様であるが、ピンのグループの大きさは等しくない。ロジックチップの数及び各々のピンの数に基づき、ピングループサイズの寸法及びセットを選択し、２のロジックチップ間に介在するロジックチップの数を最小にするとともに、各々直接隣接しているチップ対の間を充分に相互接続し、回路網がこれら二つのチップだけにつながるようにしている。相互接続のためのロジックチップをいかに構成するかを決定するには、ロジックのためのチップをいかに構成するかを決定しなければならない。ロジックチップを構成するためには：

１）基本要素変換のセクションで述べたように、設計部のロジックをロジックチップの基本要素形態に変換する。

２）ロジックチップにおけるロジック基本要素を、区分化及び配置する。ロジックチップのロジック容量内に各々適合しているサブ回路網に、設計部を区分化することに加えて、サブ回路網をお互いに対して配置し、必要とされる相互接続の量を最小とする。ゲートアレイ又は標準セルチップ自動区分化及び配置ツール（“Gate Station Reference Manual”，Mentor Graphics Corp,1987）において用いられているような、標準区分化及び配置ツール方法を用いて、いかにしてロジック基本要素をロジックチップに割り当てるかを決定し、相互接続を達成する。このことは、定評のある方法であり、ここでは、これ以上の説明を省略する。

３）ロジックチップ間の相互接続を配線する。すなわちロジックチップの中から特定のロジックチップ及びＩ／Ｏピン相互接続を選定し、ゲートアレイ又は標準セルチップ自動ルーティングツール（“Gate Station Reference” Manual”,Mentor Graphics Corp., 1987 ）のような標準ルーティングツールを用い、いかにしてチップを構成するかを決定し、相互接続を達成する。これは定評のある方法であるため、いかにしてこの方法を相互接続の問題に適用するかを除き、ここではこれ以上の説明を省略する。ロジックチップのアレイを、シングルラージゲートアレイ又は標準セルチップと同じ方法で取り扱う。各々区分化されたロジックサブネットワークは、大規模ゲートアレイロジックマクロに対応しており、相互接続されたロジックチップＩ／Ｏピンは、ルーティングに用いる配線チャンネルを規定している。特に、各々のルーティング方向には、相互接続されたロジックチップＩ／Ｏピンの各グループ毎のピンと同数のチャンネルを具えている。ロジックチップ間では多くの相互接続が可能であるので、多くのルーティング層によって、ゲートアレイのチャンネル制約を取り除くのと同様の方法を用いて、ルーティングを制約することなく、各末端部において同じチャンネルを用いる。

４）ルーティングの過剰（ルーティング処理の間或るポイントにおいてチャンネルをルーティングすることができない場合）のために相互接続を行うことが不可能な場合、調整された基準を用いて、設計部を再区分化及び／又は再配置し、過剰を除去し、再び相互接続を試みる。

５）どの回路網がどのチャンネルを使用するかについての仕様を、特定のルーティングチャンネルとＩ／Ｏピンとの間の対応に応じて、個々のロジックチップに対するネットリスト及び、ロジックチップ信号に対する特定ピンの役割に変換する。ロジック基本要素の仕様とともに、Ｉ／Ｏピン仕様及びロジックチップ内部相互接続の形態の仕様を、各ロジックチップ毎のネットリストファイルに送出する。

６）ロジックチップネットリスト変換ツールを用い、各ロジックチップ毎の構成ファイルを発生させるとともに、これらを組合せ、入力設計のための最終的なリアライザ構成ファイルを作成する。

【００１３】１．２．１．２　チャンネルルーティング相互接続

チャンネルルーティング相互接続は、直接相互接続の変形である。この場合、チップは、ロジックとしては用いられず単に相互接続を行う相互接続チップと、専らロジックとして用いられるロジックチップとに分割される。特に、ロジックチップは、お互いを直接接続するのではなく、その代わりに、ただ単に相互接続チップを接続するだけである。その他すべての点において、チャンネルルーティング相互接続は、直接相互接続方法に従って作成されている。１以上のロジックチップ及び回路網は、ルーティングチップと称する一連の相互接続チップを構成することによって相互接続する。ルーティングチップは、これらのロジックチップを接続させるとともに、お互いを接続させ、ロジックチップＩ／Ｏピン間にロジック的接続が確立される。このことは、構成可能な‘回路基板’に用いられる。チャンネルルーティング相互接続

を、一例として２次元としている：すなわち、図３に示されているように、ロジックチップは、行及び列の形態で配置されており、その周囲はルーティングチップによって完全に囲まれている。アレイを、全てがルーティングチップで構成されている行と、ロジックチップ及びルーティングチップで交互に構成されている行とで交互に構成する。このようにして、ロジックチップの周囲には、行方向及び列方向に、切目なくルーティングチップが配置されている。各チップのピンを４つのグループ、すなわち、“北側、東側、南側、西側”と称する４つのエッジに分割する。各々のチップのピンを、４つの最も近い隣接チップに格子状に接続する。：すなわち、北側のピンを北側に隣接するチップの南側ピンと接続し、東側ピンを東側に隣接するチップの西側ピンに接続する。以下同様である。このモデルは、上記例の２次元より大きな次元にまで拡張することができる。一般的には、‘ｎ’を次元の数とする場合、各ロジックチップのピンは、２＊ｎ個のグループに分割される。各ロジックチップは２＊ｎ個の隣接チップに接続している。アレイの中心においては各々のロジックチップに対して（２＊＊ｎ－１）個のルーティングチップが存在する。ロジックチップとルーティングチップとの特徴に基づき、このチャンネルルーティングモデルの一般化を同様にして用いる。ロジックチップのピンを数個のグループに分けることができる。ルーティングチップのピンも数個のグループに分けることができる。但し、ルーティングチップのピンのグループ数が、ロジックチップのピンの数と同じである必要はない。ロジックチップとルーティングチップとは、同数のピンを具えている必要はない。ロジックチップとルーティングチップとの規則的なアレイであり、且ついかなる所定のロジックチップであっても、それが最も近い隣接チップの限定セットとだけ接続されている限り、これらの変形が適用される。ロジックチップ間の相互接続を、ロジックチップを介してではなく、相互接続チップを介してのみ配線するということを除いて、直接相互接続に用いると同様の方法を用い、ロジックチップをいかに構成するかを決定するとともに、相互接続チップをいかに構成するかを決定する。回路網のロジック信号は、相互接続を完成させるのに必要なルーティングチップと同数のルーティングチップを介して流れる。各々のルーティングチップによって信号の伝搬が遅れるので、信号が流れるルーティングチップが増えれば増える程、相互接続を介しての信号伝搬遅れ時間は長くなる。ルーティングの必要条件を最小とできるように、ロジック設計部を区分化するとともに、それぞれの区分を特定のロジックチップに配置するのが一般的には望ましい。ルーティングが過剰であり、相互接続を行うことができない場合、調整された基準を用いて設計部を再区分化及び／又は再配置し、再び相互接続を行う。このサイクルは、必要な限り繰り返される。

【００１４】１．２．２　クロスバー相互接続

１．２．２．１　完全クロスバー相互接続

クロスバーとは、制約なくピンを他のピンと接続することのできる相互接続アークテクチャーである。これは、コンピュータ及び通信デバイスのスイッチング回路網において、メッセージを通信するために広く用いられている。完全なクロスバーとして構成される相互接続は、すべてのロジックチップピンに接続するとともに、いかなるピンの相互接続の組合せであっても構成可能な相互接続によって、いかなる入力設計及びロジックチップ区分化であっても、直接的に相互接続を達成することができる。その理由は、いかなるピンであっても、いかなる他のピンに直接接続することができるからである。しかし、多くのロジックチップを相互接続することのできる実用的な単一デバイスは存在しない。例えば、好適例のロジックボードは、各々接続すべき１２８個のピンを有するロジックチップを１４個具えている。合計で１７９２ピンとなり、実用的なシングルチップの容量をはるかに超えている。実用的な相互接続チップ及びデバイスからクロスバーを構成することができる。これらを構成することによって、Ｉ／Ｏピン間に任意の相互接続を実現することができる。クロスバー相互接続の場合、これらをクロスバーチップと称する。実用的なクロスバーチップからクロスバー相互接続を構成する一般的な方法は、一つのクロスバーチップを用いて、一つのロジックチップピンをクロスバーチップが有するピンと同数の他のロジックチップピンと相互接続する。図４は、わかり易くするために極めて簡略化した一例を示している。各々８個のピンを有する４個のロジックチップを相互接続する。各々９個のピンを有するクロスバーチップを用いる。３個のクロスバーチップの最も左側の列によって、ロジックチップ４のピンＨをロジックチップ１、２及び３のピンと接続する。次の列によって、ピンＧ等をロジックチップ４のピンＧに接続する。同じロジックチップに関しては、内部で接続できることから、ピンと他のピンとを接続する必要はない。クロスバーチップの隣接する８個の列は、ロジックチップ３とロジックチップ１及び２とを相互接続している。ロジックチップ４は含まれていない。その理由は、ロジックチップ４のピンを、クロスバーチップの最初の８個の列によって、ロジックチップ３のピンに接続しているからである。最後の８個の列はロジックチップ１と２とを相互接続している。合計４８個のクロスバーチップを用いる。入力設計に基づく二つの回路網は、相互接続された状態を示している。回路網Ａは、ロジックチップ１のピンＤによって駆動され、ロジックチップ４のピンＢによって受信される。１で示されているクロスバーチップは、これらのピンの両方を接続しており、ロジックチップ１のピンＤから受信し、受信したものを、チップ４のピンＢに伝達する。このようにして、ロジック接続を構成する。回路網Ｂは、

ロジックチップ２のピンＦによって駆動され、ロジックチップ３のピンＧ及びロジックチップ４のピンＧによって受信される。クロスバーチップ２は第１相互接続を行い、クロスバーチップ３は第２相互接続を行う。一般的に、必要とされるクロスバーチップの数を予測することができる。各々Ｐ１個のピンを有するＬ個のロジックチップが存在し、且つ、１個のロジックチップピンをできる限り多くの他のロジックチップピンと各々接続できるようにしているクロスバーチップがＰｘ個のピンを具えている場合：

１）ロジックチップ１の中の一つのピンを、２からＬまでのロジックチップの（Ｌ—１）Ｐ１個のピンに接続しなければならない。これには、（Ｌ—１）Ｐ１／（Ｐｘ—１）個のクロスバーチップが必要とされる。すべてのピンを接続するには、（Ｌ—１）Ｐ^{2}／（Ｐｘ—１）個のクロスバーチップを必要とする。

２）ロジックチップ２の各々のピンを、３からＬまでのロジックチップの（Ｌ—２）Ｐ１個のピンに接続しなければならない。これには、（Ｌ—２）Ｐ^{2}／（Ｐｘ—１）個のクロスバーチップを必要とする。

３）ロジックチップＬ—１の各々のピンを、ロジックチップＬのＰ１個のピンに接続しなければならない。これには、Ｐ１^{2}／（Ｐｘ—１）個のクロスバーチップを必要とする。

４）Ｘ＝（Ｌ—１）Ｐ^{2}／（Ｐｘ—１）＋（Ｌ—２）Ｐ１^{2}／（Ｐｘ—１）＋……＋Ｐ^{2}／（Ｐｘ—１）
＝（Ｌ２—Ｌ）Ｐ^{2}／２（Ｐｘ—１）

クロスバーチップの数Ｘは、ロジックチップの数の二乗とロジックチップ毎のピンの数の二乗とをかけ算したものが増加するにつれて、増加する。好適実施例のロジックボード（各々１２８個のピンを有する１４個のロジックチップ）は、各々１２９個のピンを有する１１６４８個のクロスバーチップ又は各々６５個のピンを有する２３２９６個のクロスバーチップを必要としている。クロスバー相互接続は、有用なリアライザシステムに用いるには、大規模且つ高価なものであり、非実用的である。

【００１５】１.２.２.２　完全クロスバー回路網相互接続

相互接続すべき設計回路網の数がロジックチップピンの合計数の１／２を決して超えることができないということを認識することによって、クロスバー相互接続の大きさを小さくすることができる。クロスバー回路網相互接続は、ロジック的には２つのクロスバーによって構成されており、その各々は、すべてのロジックチップピンを相互接続回路網（ＩＣＮ）と称する１セットの接続回路網に接続しており、ロジックチップピンの総数の１／２に番号を付している。１セットのロジックチップピンを、１セットのＩＣＮに接続するクロスバーチップが１セットのＩＣＮから、これらのピンへ接続を戻すこともできる（相互接続チップの一般性の撤回）ため、この相互接続をクロスバーチップで構成することもできる。各々のクロスバーチップは、１セットのロジックチップピンを１セットのＩＣＮと接続している。図５は、図４にて示したものと同一の４個のロジックチップを相互接続した一例を示す図である。各々８個のピンを有するクロスバーチップを用い、１６個のＩＣＮを設ける。３２個のクロスバーチップの各々は、４個のＩＣＮを用いて４個のロジックチップピンを接続する。回路網Ａを、クロスバーチップ１によって相互接続し、ロジックチップ１のピンＤから受信し、受信したものをＩＣＮに伝達するように構成する。また、回路網Ａを、クロスバーチップ２によって相互接続し、前記ＩＣＮから受信し、ロジックチップ４のピンＢを駆動する。このようにしてロジック接続を確立する。回路網Ｂは、ロジックチップ２のピンＦによって駆動され、クロスバーチップ４を介してロジックチップ３のピンＧで受信されるとともに、クロスバーチップ５を介してロジックチップ４のピンＧで受信される。好適実施例のロジックボード（各々１２８個のピンを有する１４個のロジックチップ）のクロスバー回路網相互接続は、各々１２８個のピンを有する３９２個のクロスバーチップ、又は各々６４個のピンを有する１５６８個のクロスバーチップを必要とする。クロスバー回路網相互接続では、使用するクロスバーチップの数は単なるクロスバーよりも少ない。クロスバー回路網相互接続の大きさは、ロジックチップの数と、ロジックチップピンの総数との積が増加するにつれて、大きくなる。これは、ロジックチップ数の二乗に達する。これは、純粋なクロスバーよりは優れているも、依然として望まれる直接スケーリングではない。

【００１６】１.２.２.３　部分的クロスバー相互接続

ロジックチップそれ自体によって、クロスバーが開発できない付加的な自由度を提供することができる。その理由は、ロジック回路網の所定の入力又は出力を、いかなるＩ／Ｏピンをも用いることができるように構成することができるからである。すなわち、特定の回路網とは無関係に構成するからである。この自由度によって部分的クロスバー相互接続をすることができる。これが、自由度をロジックチップの定義中に明示している理由である。部分的クロスバー相互接続では、各ロジックチップを分割するのと同様にして、各ロジックチップのＩ／Ｏピンを適切なサブセットに分割する。各クロスバーチップのピンを、各ロジックチップの各々から同じピンのサブセットに接続する。このようにして、クロスバーチップ‘ｎ’を、各ロジックチップのピンのサブセット‘ｎ’に接続する。サブセットと同数のクロスバーチップを用いる。各々のクロスバーチップは、サブセットのピンの数とロジックチップの数とをかけ算した数と同数のピンを有している。各ロジックチップ／クロスバーチップ対を、各サブセット中のピンと同数のバスと称するワイヤで相互接続する。各クロスバーチップを各ロジッ

クチップのピンと同一のサブセットに接続しているため、一つのロジックチップのピンにおける一つのサブセット中のＩ／Ｏピンから、もう一つのロジックチップのピンにおける別のサブセット中のＩ／Ｏピンへの相互接続を構成することはできない。このことは、相互接続すべき各々のロジックチップのピンの同一のサブセットから、Ｉ／Ｏピンを用い、各々の回路網を相互接続し、適宜にロジックチップを構成することによって避けられる。回路網に接続されるロジックチップ中に構成されるロジックチップに割り当てることのできるいかなるＩ／Ｏピンを用いても、ロジックチップを構成できるようにするため、一方のＩ／Ｏピンは、他方のＩ／Ｏピンと同様のものである。一般的なパターンを図６に示す。この図において、ロジックチップとクロスバーチップとを接続している各々のラインは、ロジックチップピンのサブセットを示している。各クロスバーチップをすべてのロジックチップのピンのサブセットに接続する。逆に言えば、このことは、各ロジックチップをすべてのクロスバーチップのピンのサブセットに接続していることを示している。これらの例では、クロスバーチップの数がロジックチップの数と等しい必要はない。好適な実現例では、このようなことは言えない。図７は、図１及び図２と同一の４個のロジックチップを相互接続している例を示している。各々８個のピンを有する４個のクロスバーチップを用いる。各クロスバーチップは、各ロジックチップにおいて同一の２個のピンを接続している。クロスバーチップ１をロジックチップ１～４の各々のピンＡ及びＢに接続する。クロスバーチップ２をすべてのピンＣ及びＤに接続し、クロスバーチップ３をすべてのピンＥ及びＦに接続するとともに、クロスバーチップ４をすべてのピンＧ及びＨに接続する。前記例の設計回路網Ａでは、ロジックチップ４のピンＢにおいて受信が行われるが、回路網Ａをロジックチップ１のピンＤにおけるドライバに相互接続することのできるクロスバーチップは設けられていない。いかなるＩ／Ｏピンであっても、回路網Ａを受信するロジックチップ４において構成されるロジックに割り当てることができるので、ピンＣはピンＢと同様であり、これを、他の回路網に用いることができる。結果的に、回路網Ａは代わりにピンＣによって受信され、クロスバーチップ２を構成することで、相互接続を達成する。設計回路網Ｂは、ロジックチップ３のピンＧ及びロジックチップ４のピンＧによって受信されるが、この回路網Ｂを、ロジックチップ２のピンＦにおけるドライバと相互接続できるクロスバーチップは設けていない。回路網Ｂは、代わりにピンＨによって駆動され、クロスバーチップ４を構成することで相互接続を達成する。好適な実施例では、部分的クロスバー相互接続を用いている。このロジックボードは、各々１２８個のピンを有する１４個のロジックチップを具えており、各々５６個のピンを有する３２個のクロスバーチップによって相互接続されている。ロジックチップピンを、各々４個のピンを有する３２個の適切なサブセットに分割するとともに、各クロスバーチップのピンを、各々４個のピンからなる１４個のサブセットに分割する。クロスバーチップ‘ｎ’を各ロジックチップピンのサブセット‘ｎ’に接続し、各ロジックチップ／クロスバーチップ対を４個のパスによって相互接続する。すべてのクロスバー相互接続の中で、部分的クロスバーの使用するクロスバーチップの数は最小である。部分的クロスバーのサイズは、ロジックチップピンの総数が増大するに従って直接的に増大する。このことは、ロジックチップの数、更にはロジック容量に直接関連するものであり、望ましい結果である。これを使用するのは比較的容易なことである。その理由は、部分的クロスバーが規則的であり、そのパスをテーブルで表現することが可能であり、更に特定の相互接続をいかにして決定するかは、単にパスの最適なペアをテーブルで捜すだけだからである。

【００１７】１.２.２.４　部分的相互接続の機能

部分的なクロスバー相互接続は、完全クロスバーが処理できるのと同数の回路網を処理することはできない。ソースロジックチップにおいて、他の回路網に対して一つだけ用いられていないＩ／Ｏピンが、行先ロジックチップに至るパスが同様にすべて使用されているクロスバーチップとつながっている場合、部分的クロスバー相互接続は回路網を相互接続することができない。行先ロジックチップは利用可能なピンを有しているが、このような場合、Ｉ／Ｏピンはソースピンがすべて使用されている他のクロスバーにつながっており、これらのクロスバーから最初へ戻る途はない。部分的なクロスバー相互接続の容量はそのアーキテクチャーに依存している。一つの極端な例では、一つのロジックチップピンサブセットだけが存在し、一つのクロスバーがすべてのピンに作用する。このような装置は、最大の相互接続能力を有するが、非現実的な完全クロスバー接続である。他の極端な例では、サブセットサイズは、ロジックチップのピンと同数のクロスバーチップを有するものである。これは、すべての部分的クロスバーを相互接続する能力は最小であるが、依然として充分な能力を有している。極端な例の間では、各クロスバーチップが、２、３又はそれ以上の各ロジックチップのピンに作用するアーキテクチャーとなっている。クロスバーチップ数が減少し、クロスバーチップ毎のピン数が増加するにつれて、より多くの相互接続能力が利用可能となる。この変更は、以前より注目されていることではあるが、種々のクロスバーチップが作用するために、相互接続することのできない未使用ロジックチップが存在するということによるものである。クロスバーチップの数がより少なくなるとともに、幅が広くなるにつれて、このような変更は、一般的には生じなくなる。完全クロスバーは、すべてのピンを定義されたいかなるパターンにも相互接続することができ

る。他の簡単な一例として、各々３個のピンを有する、参照番号１、２及び３を付した３個のロジックチップが存在し、且つ、４個の回路網Ａ、Ｂ、Ｃ及びＤが存在するものと仮定する。回路網Ａはロジックチップ１及び２を接続し、回路網Ｂはロジックチップ１及び３を接続し、回路網Ｃはロジックチップ２及び３を接続し、回路網Ｄはロジックチップ１及び２を接続する。図８ａ及び８ｂにおいて、各ロジックチップのピンをセルの行として示しており、各クロスバーチップはクロスバーチップが作用するピン数と同数の列をカバーしている。第１のケース（図８ａ）では、各々１つのピンの幅を有する参照番号１、２及び３で示される３つのクロスバーチップを使用する。各クロスバーチップは、ただ１つの回路網を接続できるにすぎない。すなわち、クロスバーチップ１は、回路網Ａを相互接続するようプログラムされており、クロスバーチップ２は回路網Ｂを接続し、クロスバーチッ３は回路網Ｃを接続する。未使用ロジックチップピンを利用することもできるが、回路網Ｄは未接続のままである。第２のケース（図８ｂ）では、３個のピン幅を有する完全クロスバーを、クロスバーチップ１、２及び３の代わりに用いて回路網Ｄを接続することができる。種々の部分的クロスバー相互接続アーキテクチャによって相互接続することのできる入力設計回路網の数に基づき、アナリシス及びコンピュータモデル化を行う。結果的には、ナロウな部分的クロスバーは、ワイドなもの又は完全クロスバーと、ほぼ同じ程度に効果的である。例えば、好適実現例（１４個の１２８ピンロジックチップ、３２個の５６ピンクロスバーチップ）のロジックボードに用いられている相互接続は、完全クロスバーの有する相互接続容量の９８％を示している。モデリングにおいて想定されているように、実際の入力設計部が、利用可能なマルチロジックチップ回路網及びロジックチップピンの数を最大限に必要とすることは極めて稀である。実際の設計部は、ほぼ常に最大限よりも少ない回路網を有しており、上述のモデルの部分的クロスバーによって接続される回路網の平均個数よりも少ない回路網、通常かなり少ない回路網を有している。このことは、ロジック容量を保持するのに絶対的に必要であるより多くの、小さな比率のロジックチップピン及びクロスバーチップを用いることで保障され、このようにしてナロウな部分的クロスバーによって、実際の設計部がほとんど常に相互接続可能であることを保障している。ナロウなクロスバーチップは、ワイドなクロスバーチップよりかなり小さく、それ故、ピン単位では高価なものではない。

【００１８】１.２.３　相互接続トライステート回路網
部分的クロスバー相互接続のようなアクティブ相互接続と実際のワイヤのようなパッシブ相互接続との重要な相違は、アクティブ相互接続が無方向性であるということである。実際、各々の相互接続は、チップ境界において金属及びトレースによって結合する一連のドライバ及びレシーバを具えている。通常の回路網は一つのドライバを有し、アクティブ相互接続で固定されたドライバ及びレシーバを用いて作成される。実際に設計する回路網の幾つかはトライステートであり、図９に示すような幾つかのトライステートドライバを有している。任意の所定時間において、最大で１個のドライバが活動状態であり、その他のドライバは回路網に対して高インピーダンスの状態にある。（伝播遅延を無視すると）すべてのレシーバは常に同一のロジックレベルにある。

【００１９】１.２.３.１　トライステート回路網を積の和に置き換える
全回路網を同一のロジックチップへと区分化する場合、回路網を、積の２ステート加算、すなわち、図１０にて示すような、等価なマルチプレクサで置き換えることができる。アクティブイネーブルが存在しない場合、この回路網は低ロジックレベルを出力する。時々、トライステート回路網は、受動的に高ロジックレベルにされる。必要ならば、各ＡＮＤゲートへのデータ入力を反転するとともに、最終加算ゲート出力を反転することで、イネーブルできない場合、積の和は高ロジックレベルを出力する。１以上のイネーブルがアクティブの場合、結果はすべての入力信号の加算（ＯＲ）となる。このことは容認される。その理由は、異なるデータで１以上がイネーブルされる場合、実際のトライステートドライバの動きを規定していないからである。図１１ａ及び１１ｂは、２種類の回路網：すなわち“フローティングハイ（floating high）”及び“フローティングロー（floating low）”を示している。リアライザシステムの設計変換システムの基本要素変換部分は、和又は積の置き換えを行う。その理由は、好適実現例のロジックチップ及びクロスバーチップとして用いられるXilinx ＬＣＡが、すべての回路網におけるトライステート駆動を一様に維持していないからである。トライステートドライバは、ＬＣＡの境界におけるすべてのＩ／Ｏピンを利用することができる。XC3000シリーズＬＣＡの内部で利用できるトライステートドライバの数は制限されており、チップ間を結んでいる内部相互接続の数が小さいことから、各ドライバはＣＬＢの一つの行にだけ作用する。トライステート回路網をこれらの相互接続に作成することによって、分割化に他の制約が加わり、ＬＣＡにおけるＣＬＢの配置の自由度を制約することとなる。同時に、回路網毎に少数のドライバとトライステート接続することは、ある種のゲートアレイライブラリセルにおいては一般的なことである。結果的に、このように複雑となることを避けられる場合には、積の和の置き換えを行う。設計部を多重ロジックチップに分割化することにより、トライステート回路網を２以上のロジックチップに亘って分割する場合、積の和を局所的に用いて、ロジックチップと回路網との各々の接続をロジックチップ境界における単

ーのドライバ及び／又はレシーバに引き下げる。図１２は、２つのドライバ及び２つのレシーバを一緒に示している。２つのドライバは局所的な積の和によって構成され、このようにして、単一のドライバ接続のみを要件として積の総和を与える。同様にして、単一のレシーバ接続を、２つのレシーバに亘って構成する。このようにして、アクティブ相互接続がなされる。トライステート回路網におけるいかなる所定の点においても、駆動“方向”はどのドライバを活動状態とするかに依存している。このことは、パッシブ相互接続と何ら差異はないが、アクティブ相互接続では、能動的に正しい方向に駆動及び受信が行われるように、アクティブ相互接続を構成しなければならない。構成によっては、このことを部分的クロスバー相互接続によって達成することができる。

【００２０】１.２.３.２　ロジック加算構成

３つの構成は、回路網を積の和に引き下げることに基づいている。ロジック加算構成は、図１３に示されているように、加算ＯＲゲートを、関連するロジックチップ中に配置する。積を発生させるＡＮＤゲートを駆動ロジックチップで構成する。この駆動チップの各々は出力チップを必要としている。各受信ロジックチップは入力ピンを必要とし、特別な場合、加算ロジックチップは各ドライバ用の入力ピンと出力ピンとを必要とする。これらの接続はすべて無方向性であり、各チップの境界に亘ってＯＢＵＦ／ＩＢＵＦ対を具えている。ドライバのピンが高価であるので、駆動ロジックチップを加算チップとして選択する必要がある。簡単のため、図中には関連するＬＣＡ基本要素をすべて示してはいない。駆動入力ピンから受信出力ピンに至る実際のパスは、ドライバのＣＬＢ及びＯＢＵＦと、クロスバーのＩＢＵＦ／ＯＢＵＦと、加算チップのＩＢＵＦ、ＣＬＢ及びＯＢＵＦと、クロスバーの他のＩＢＵＦ／ＯＢＵＦと、レシーバのＩＢＵＦとを具えている。クロスバーＩＢＵＦ遅延をＩｘとし、ロジックＣＬＢ遅延をＣ１等とした場合、全データ通路遅延は、Ｃ１＋Ｏ１＋Ｉｘ＋Ｏｘ＋Ｉ１＋Ｃ１＋Ｏ１＋Ｉｘ＋Ｏｘ＋Ｉ１である。特別な場合、すなわちロジックチップをＸＣ３０９０－７０とし、クロスバーをＸＣ２０１８－７０とした場合、遅延の総計の最大は、８２ｎｓに内部ＬＣＡ相互接続遅延を加えたものに等しい。同じ遅延がイネーブルにも当てはまる。ｎビットバスを相互接続する場合、バスの各ビットに対してすべてのイネーブルは同様のものである。この特別な構成において、駆動チップ中に積のゲートを設け、イネーブルを内部に設け、バスに必要なピンを１ビットの場合のピン数のちょうどｎ倍とする。

【００２１】１.２.３.３　クロスバー加算構成

クロスバー加算構成において、加算ＯＲゲートをクロスバーチップに配置する。この場合、図１４に示されているようなロジックを利用することのできるＬＣＡのようなＥＲＣＧＡを用いて、いくつかの例のクロスバーチップを具体化している。各ロジックチップは、ドライバとしての１ピン及び／又はレシーバとして１ピンを必要としている。クロスバーチップは、加算ゲートのための１以上のロジック素子を有している必要がある。クロスバー加算とは、ロジックチップ中のロジックをすべて用い、クロスバーチップ中のロジックを全く用いず実行するということではない。重要な相違点は、クロスバーチップに配置されたロジックが、実現される設計ロジックの一部ではないということである。ロジックは、単に、トライステート回路網の相互接続機能を達成する役割を果たすに過ぎない。この構成では、２以上の駆動ロジックチップを設けた場合、従来よりも使用するピンの数が少ない。ｎビットバスはピンのｎ倍も作用する。全遅延は、Ｃ１＋Ｏ１＋Ｉｘ＋Ｃｘ＋Ｏｘ＋Ｉ１、すなわち、最大５１ｎｓにまで引き下げられる。イネーブルも同じ遅延を有する。

【００２２】１.２.３.４　双方向性クロスバー加算構成

図１５にて示されているように、クロスバーチップの加算ゲートを、双方向性クロスバー構成における双方向性接続を介して連絡している。ＯＲゲートへのバスをイネーブルできるＡＮＤゲートを、クロスバーチップ中に設け、フィードバックラッチアップバスをブロック化する。ロジックチップは、レシーバのみの場合には１つのピンを必要とし、ドライバ又は、レシーバ及びドライバ双方の場合には２つのピンを必要とする。この２つのピンの一方は信号自体のためのものであり、もう一方はイネーブル出力のためのものであり、これはクロスバーチップに用いられる。１ビット以上の信号イネーブルを用い、マルチビットバスによって、相互接続を減少させることができる。同一のクロスバーチップを介して、１ビット以上のバスを相互接続する場合、１セットのイネーブル信号をチップに供給する必要がある。好適なＬＣＡ例においては、全データ通路遅延を、Ｏ１＋Ｉｘ＋Ｃｘ＋Ｏｘ＋Ｉ１、すなわち、４２ｎｓとしている。積の和が、２以上のＣＬＢを用いる場合、付加的なＣｘ（１０ｎｓ）を加えることができる。イネーブル遅延は出力遅延Ｏ１ではなくて、ＯＢＵＦＺのイネーブル遅延Ｅ１に依存している。

【００２３】１.２.３.５　双方向性クロスバートライステート構成

これまで説明したすべての構成を同一のハードウエアで使用することができることに注意しなければならない。基本要素の配置及び相互接続のみが変化する。最終的に、クロスバーチップが内部トライステートを維持する場合、図１６にて示すように、双方向性クロスバートライステート構成によって、クロスバーチップ内部の実際のトライステート回路網が二重になる。各ロジックチップの実際のトライステートドライバは、クロスバーチッ

プのバスにそのまま伝えられる。このことは、イネーブル信号の相互接続によって達成されるはずである。ドライバがイネーブルされていない場合、クロスバーチップのバスを駆動させる。ＬＣＡをクロスバーチップとして用いる場合、上記内部トライステート相互接続を用いる。特に、ロジックチップの境界にＩＢＵＦ／ＯＢＵＦＺ対、クロスバーチップ境界に各ロジックチップの他のＩＢＵＦ／ＯＢＵＦＺ対、各ロジックチップにＴＢＵＦを設け、内部トライステートラインを駆動する。各イネーブルはＯＢＵＦ及びＩＢＵＦを通過する。イネーブルされたデータパス遅延の総計は、Ｏ１＋Ｉｘ＋Ｔｘ＋Ｏｘ＋Ｉ１、すなわち、３９ｎｓ（ＸＣ３０３０—７０ＬＣＡクロスバー）であり、イネーブル遅延の総計は、Ｏ１＋Ｉｘ＋ＴＥｘ＋Ｏｘ＋Ｉ１、すなわち４５ｎｓである。以前のように、２ビット以上のバスを、同一のクロスバーチップを介して相互接続する場合、１セットのイネーブル信号のみを、チップに供給する必要がある。この構成では、クロスバーチップをＬＣＡ又は内部トライステート機能を有するＥＲＣＧＡとする必要があり、これら内部相互接続の利用を条件としている。特に、XC2000シリーズは、内部トライステートを有していないが、XC3000パーツは有している。XC3030は、８０個のＩ／Ｏピン、１００個のＣＬＢ及び２０個のトライステート駆動可能内部‘ロングライン’を有している。このようにして、最大で２０個のこのようなトライステート回路網を、この構成中の一つのクロスバーチップによって相互接続し得る。これは、相互接続の限界となり得るが、いろいろな場合のほんの一部にすぎず、Ｉ／Ｏピンの限界を与えるものである。現在のところ、XC3030はXC2018の２倍高価である。ハードウエアにトライステート構成を用いる場合、他の構成は妨げとはならず同様に使用することができる。

【００２４】1.2.3.6　すべての構成の概要

このチャートは、構成を要約したものである。

	ロジック加算	クロスバー加算	双方向性クロスバー加算	双方向性クロスバートライステート
ピン／ロジックチップ双方向性	＝駆動＋受信	2	1データパス1共有可能イネーブル	1データパス1共有可能イネーブル
駆動のみ	第1チップ：0 その他：2	1	1データパス	1データパス
受信のみ	第1非加算：2 その他：1	1	1共有可能イネーブル	1共有イネーブル

遅延

（ＬＣＡクロスバーチップ：＋ＬＣＡ相互接続、７０ＭＨｚ　ＬＣＡチップスピードであると仮定する）

データパス	82ns	51	42	39
イネーブル	82	51	46	45

チップ毎のリソース

（ｄはドライバの数）

駆動のみ	AND中に2個	AND中に2個	0	0
受信のみ	0	0	0	0
双方向性	AND中に2個	AND中に2個	0	0
クロスバー	0	OR中にd個	OR中にd個	d個のTBUF

明らかに、ロジック加算構成は有効ではない。クロスバー加算は、かなり高速で、少数のピンを使用し、多くの場合シンプルである。双方向性クロスバー加算は、依然わずかに高速であり、双方向性バスのピン数を減少させ

る可能性を有しているが、かなり複雑であり、クロスバーチップに限られたロジックリソースをより必要とする。トライステート構成によって、同様のピンを必要とし、及び遅延が生じるが、より高価なクロスバーチップを必要とする。

【００２５】１.２.３.７　普通のクロスバー加算構成と双方向性クロスバー加算構成との比較

最も効果的な構成の特性をテストすることは有益である。以下の表は、普通及び双方向性のクロスバー加算構成を用い、多数の双方向性回路網を相互接続し、且つ、ＬＣＡをクロスバーチップとして用いる場合に生じるクロスバーＣＬＢの数とクロスバーＣＬＢ遅延とを示している。７２個のＩ／Ｏピンを有し、１００個のＣＬＢを用いることのできる、XC2018-70クロスバーチップを用いるものと仮定する。各々のＣＬＢは４個までの入力端子及び２個までの出力端子をサポートする。各ロジックチップがイネーブルを共有せず、回路網と双方向性の接続を有し、各テストにおいて、クロスバーチップの７２個のＩ／Ｏピンすべてを用いるものと仮定する。

	クロスバー加算	双方向性クロスバー加算
１８個の双方向性回路網サーピング	９　ＣＬＢｓ	１８　ＣＬＢｓ
２個のロジックチップの各々	１　Ｃｘ	１　Ｃｘ
１２個の双方向性回路網サーピング	１２　ＣＬＢｓ	２４　ＣＬＢｓ
３個のロジックチップの各々	１　Ｃｘ	２　Ｃｘ
９個の双方向性回路網サーピング	９　ＣＬＢｓ	２７　ＣＬＢｓ
４個のロジックチップの各々	１　Ｃｘ	２　Ｃｘ
６個の双方向性回路網サーピング	１２　ＣＬＢｓ	２４　ＣＬＢｓ
６個のロジックチップの各々	２　Ｃｘ	２　Ｃｘ
３個の双方向性回路網サーピング	１２　ＣＬＢｓ	３０　ＣＬＢｓ
１２個のロジックチップの各々	２　Ｃｘ	３　Ｃｘ

双方向クロスバー加算構成はＣＬＢを２．５倍まで使用し、クロスバーチップがルートしない可能性、すなわち内部相互接続遅延が大きくなる可能性が増大する。しかし、依然として１００個のＣＬＢが使用可能となるまでには程遠い。代わりに、無方向性構成では、ロジックチップを特別なゲートを操作するのに好適な位置に設けるが、ロジックチップにかなり多くのゲートを設けている。双方向性構成では、特別なＣｘ遅延がしばしば生じそのスピードの利点を相殺してしまう。リアライザシステムの好適例では、トライステート回路網のためのクロスバー加算構成を用いている。

【００２６】１.２.４　システムレベル相互接続

クロスバーチップによって相互接続された１セットのロジックチップのパッケージ化は、単一の回路ボードにおいて行うのが一般的である。システムが大規模すぎて単一のボードに適合しない場合には、システムレベル相互接続を用いて、ボードを同じように相互接続しなければならない。極めて広域のバス配線のために、２以上の回路ボードに亘る単一の部分的クロスバー相互接続及びロジックチップを拡張することは非実用的である。例えば、３２個の１２８ピンロジックチップと６４ピンのクロスバーチップとの複合体を、２つのボードに、それぞれ１６個のロジックチップと３２個のクロスバーとに分割するものと仮定する。複合体を、ロジックチップとクロスバーチップとの間で切断する場合、背面接続の対を介して、ロジックチップとクロスバーチップとの間に総計で４０９６個の相互接続を行う必要がある。これとは別の方法で、‘中間’で、すなわち、１６個のロジックチップ及び３２個のクロスバーチップで各ボード毎に切断する場合、ボード１のロジックチップをボード２のクロスバーに接続するバス（１６個のロジック×６４個のピン＝１０２４）及びその逆のバス（もう一つの１０２４、合計で２０４８）の全てをクロスさせなければならない。このような単一の相互接続では発展の可能性がないといった他の制約もある。定義によれば、各クロスバーチップは全てのロジックチップと接続している。特定数のロジックチップで構成する場合、それ以上を加えることができない。その代わりに、回路基板上に一緒にパッケージ化することのできるロジックチップとクロスバーチップとの最大規模の複合体をロジックボードと称し、モジュールとして用い、これらの多数をシステムレベル相互接続によって接続する。２以上のボードに及ぶ相互接続回路網を提供するために、各ロジックボードのクロスバーチップの各々の付加的なＩ／Ｏピンに対して、ボードから離れた付加的な接続を行い、ロジックボードＩ／Ｏピンを確立する（図１７）。ロジックボードＩ／Ｏピンに接続するのに用いられるクロスバーチップＩ／Ｏピンは、ボードのロジックチップＩ／Ｏピンと接続しているものとは別のものである。

【００２７】１.２.４.１　部分的クロスバーシステムレベル相互接続

ロジックボードを相互接続するための一つの手段では、部分的なクロスバー相互接続を再び適用し、各ボードをロジックチップの如く取り扱うとともに、付加的なクロスバーチップのセットを用いてボードのＩ／Ｏピンを相互接続する。この部分的なクロスバーは、ボックス中の

すべてのボードを相互接続する。第３番目の相互接続を、ラック中のすべてのボックス等を相互接続することに再び適用する。終始同一の相互接続方法を適用することによって、概念の簡易化及びボードレベル相互接続との一体化といった利点が得られる。リアライザシステム中のクロスバーチップを区別するために、ロジックチップを相互接続している部分的クロスバー相互接続をＸレベル相互接続と称し、そのクロスバーチップをＸチップと称する。ロジックボードを相互接続している相互接続をＹレベル相互接続と称し、そのクロスバーチップをＹチップと称する。Ｘレベル相互接続では、各ロジックボードの分割と同様の分割を用いて、各ロジックボードのＩ／Ｏピンを適切なサブセットに分割する。各Ｙチップのピンを、すべてのロジックボードの各々からのピンの同一のサブセットに接続する。サブセットと同数のＹチップを使用する。各Ｙチップは、サブセットのピン数とロジックボードのピン数とをかけ算した結果と同数のピンを有している。同様にして、各Ｙチップの付加的なＩ／Ｏピンにボックスから離れた付加的な接続を行い、ボックスＩ／Ｏピンを構成する。この各々を、各ボックスにおける分割と同様の分割方法を用いて適切なサブセットに分割する（図１８）。各Ｚチップのピンを、各ボックスからのピンの同一のサブセットに接続する。サブセットと同数のＺチップを用いる。各Ｚチップは、サブセットのピン数とボックスの数とのかけ算した結果と同数のピンを有している。部分的なクロスバー相互接続の付加的なレベルを構成する方法を、必要である限り継続する。入力設計部を区分化する場合、ロジックチップのＩ／Ｏピンの数が限定されているように、ボード上及びボードから離れて配線されている回路網がボードのＩ／Ｏピンを介しており、ボードのＩ／Ｏピンの数の限定には一定の制約があることがわかる。多重ボックスリアライザシステムにおいて、ボックスＩ／Ｏピンの数が限定されていること等がわかる。設計メモリのような特別の機能が付随する場合を除き、チップ、ボード又はカートリッヂに関する配置を最適にするための相互接続シンメトリー手段は必ずしも必要ではない。双方向性回路網及びバスを、クロスバー加算方法のようなトライステートセクションにおいて説明した方法のうちの一つを用いて具体化する。この方法は、回路網がつながっている相互接続階層の各レベルに亘って適用される。好適な具体例は次のとおりである。

・全ハードウエアシステムに亘る三つのレベルにおいて、部分的クロスバー相互接続を階層的に用いる。

・ロジックボードが、各々１２８個の相互接続されたＩ／Ｏピンを有する、最大１４個のロジックチップと３２個のＸチップから成るＸレベル部分的クロスバーとを具えている。各Ｘチップは、各々１４個（全５６個）のＬチップへつながっている４個のバスと２個のＹチップの各々へつながっている８個のバスとを具え、全体として、ボード毎に５１２個のロジックボードＩ／Ｏピンを具えている。

・１個のボックスが、各々５１２個の相互接続されたＩ／Ｏピンを有する１～８個のボードと６４個のＹチップから成るＹレベル部分的クロスバーとを具えている。各Ｙチップは、ロジックボードＩ／Ｏピンを介して各ボードのＸチップにつながっている８個のバスと１個のＺチップにつながっている８個のバスとを具えており、合計でボックス当り５１２個のボックスＩ／Ｏピンを具えている。

・ラッチは、各々５１２個の相互接続Ｉ／Ｏピンを有する。１～８個のボックスと６４個のＺチップから成るＺレベルクロスバーとを具えている。各Ｚチップは、ボックスＩ／Ｏピンを介して、各ボックス中のＹチップにつながっている８個のバスを具えている。

【００２８】１.２.４.２　双方向性バスシステムレベル相互接続

コンピュータハードウエアの実行には、双方向性バスの背面を用いての、ロジックボードのシステムレベル相互接続に関する他の方法が必要となる。以前と同様、各ロジックボードにＩ／Ｏピンを設け、各ボードのＩ／Ｏピンを、バスワイヤによって、他のすべてのボードの同じＩ／Ｏピンに接続する（図１９）。いくつかのロジックボードＩ／Ｏピンは無駄である。すなわち、設計回路網に対して相互接続不能である。その理由は、一つの設計回路網を相互接続するためのバスワイヤを使用することによって、バスを共有している他のすべてのボードのバスワイヤに接続されたピンを使用できなくしてしまうからである。相互接続することのできる設計回路網の最大数は、バスワイヤの数、すなわち、ボード毎のＩ／Ｏピンの数と等しい。特別の場合として、８個のボードにおいて、一つの共通相互接続バスを、各ボードの５１２個のＩ／Ｏピンを接続している５１２個のバスワイヤが共有している（図２０）。２番目、３番目、４番目、５番目、６番目、７番目及び８番目ボードの回路網が異なる配線であると仮定すると、解析により、各ボードと接続している回路網の平均数は各々の場合５１２であり、すべての回路網において、ボード及びバスは１１６６個のピン幅まで許容されるはずであることがわかる。このことは、単一の背面のボード数を小さくし続けることによって部分的に軽減される。しかし一組の双方向性バスと相互接続されたボードの最大数は制限されている。大規模システムをより効果的に構成するためには、バスのグループを階層的に相互接続する。図２１に示されている第１の例では、各々４個のボードを接続している２組のバスＸ０及びＸ１を有している。Ｘレベルのバスを、他のバスＹで相互接続する。Ｘバス中の各々のワイヤを、再構成可能な双方向性トランシーバによってＹの片方に接続する。双方向性トランシーバの構成によって、Ｘ及びＹのワイヤが絶縁されているかどうか、ＸがＹを駆動

又はＹがＸを駆動するかどうかが決定される。回路網が、左側のボードの組又は右側のボードの組のみを接続する場合、Ｘレベルバスの一方又は他方のみを用いる。両側にボードを具えている場合、Ｘ０及びＸ１のワイヤを各々使用し、これらのワイヤをトランシーバを介してＹのワイヤで相互接続する。各ボードは、Ｘレベルバスの一方の幅と同数のＩ／Ｏピンを有している必要がある。Ｙを介しての相互接続が双方向性、すなわち、Ｘ０又はＸ１のいずれか一方によって駆動される場合、追加的な信号がＸ０及びＸ１から流れ、トランシーバの方向性を動的に制御する。この相互接続を分析しボード間の回路網を相互接続する機能を示す。この際、上記と同じ回路網ピン数及びＩ／Ｏピン数であると仮定する。シングルレベル方法では、全回路網の総計と同じ幅を必要とするが、これを２つに分割し、必要とされる最大幅を１０％～１５％に減少させている。階層は、最大でも、バス当りただ２つのボード又は２つのグループのボードを有するのみである（図２２）。双方向性バス相互接続は簡素であり、組立てが容易であるが高価である。その理由は、かなり多くのロジックボードＩ／Ｏピンを他のボードの回路網に接続することによって無駄にしているからである。このことを避けるために、階層化及び短絡背面を導入しても、効果が極めて小さいことが証明されている。更に、双方向性トランシーバを導入することによって、シングルレベル背面バス相互接続が部分的クロスバーよりも優位にあるスピード及びコストの面での利点を除去してしまう。結果的に、好適例のシステムレベル相互接続に部分的クロスバーを用いる。

【００２９】１．３　特定目的の構成素子

特定目的の構成素子とは、入力設計を実現し、好適例のロジックボードのＬチップの位置に取り付けるハードウエア構成素子であるが、ロジックチップを構成する組合せロジックゲート又はフリップフロップではない。

【００３０】１．３．１　設計部メモリ

多くの入力設計部はメモリを具えている。ロジックチップがメモリを具えているならば理想的である。電流ロジックチップデバイスはメモリを具えていない。メモリを具えるとするとメガバイト規模のメインメモリを依然として必要とし、これは、ロジックチップには決して望めないことである。従って、設計メモリデバイスを、リアライザシステム中に設けることとする。

【００３１】１．３．１．１　設計部のメモリアーキテクチャ

設計部メモリモジュールのアーキテクチャを以下の要件に基づき構成する。：

ａ）設計部のメモリモジュールは、設計部の一部であるため、他の構成要素と自由に相互接続できるようにする必要がある。

ｂ）ロジックチップと同様に効果的な相互接続を行うことができるように、データ、アドレス及び制御入出力の割り当てに自由度を設け、バスの相互接続を行う必要がある。

ｃ）種々の容量及びビット幅を有する１以上の設計部メモリを実現できる構成の変更を可能にする必要がある。

ｄ）ホストインタフェースが、設計部とデバッガタイプの対話ができるように、アクセス可能である必要がある。

ｅ）メモリモジュールはダイナミックではなくスタティックである必要がある。これによって、設計部を意のままに、ストップ、スタート又は任意のクロックスピードでランさせることができる。これらの要件を満足するメモリモジュールの一般的アーキテクチャーを図２３に示す。

設計部との相互接続可能性及びリアライザシステムの物理的構成に関する柔軟性を維持するために、置き換えたロジックチップと同一の相互接続及び他のピンに接続されたＬチップソケットにプラグで接続するように、メモリモジュールを設計する。必要なだけのモジュールを取り付ける。ＲＡＭチップを相互接続に直接接続しない。その理由は、主にチップのデータ、アドレス及び制御機能を特定のピンに定めているからである。部分的クロスバー相互接続の成功が、自由にＩ／Ｏピンとの内部相互接続を割り当てることのできるロジックチップの機能に依存しているために、ロジックチップの場所に配置されたノンロジックチップデバイスは、同様の機能を有している必要がある。このことを達成するとともに、メモリモジュールに他のロジック機能を提供するために、ロジックチップをメモリモジュール中に取り付け、ＲＡＭチップをクロスバーのＸチップと相互接続する。特定のＲＡＭピンを、任意に選択されたＸチップピンと相互接続し、メモリモジュールを構成する。この際、メモリモジュールを使用する場所に、ロジックチップが使用するのと同一のＬ—Ｘバスを使用する。１個よりも多くのロジックチップをモジュール毎に使用する。その理由は、接続すべきＲＡＭピン及びＬ—Ｘバスの数が多いからである。メモリモジュールのロジックチップによって、メモリモジュールに構成可能性及びホストアクセス可能性を提供する。ロジックチップを介して、種々の容量、ビット幅及び入力／出力構造を有するＲＡＭチップを接続するように、アドレス、データ及び制御バスを構成する。メモリモジュールを、１個の大規模メモリ又は幾つかの小規模メモリで構成することができる。これらのロジックチップの各々をホストインタフェースバスに接続するとともに、バスインタフェースロジックをロジックチップ中に構成することによって、ホストプロセッサが、ＲＡＭをランダムにアクセスすることができる機能を実現する。これによって、デバッガのようなコンピュータプログラムを用いて、メモリ内容を検査及び修正する。これらのロジック構造の具体例を、以下に示す。実現する設計部のタイミングに関する要件を満足する、入手可能

で高密度且つ安価なスタティックメモリを、設計部メモリとして選択する。好適例では、このようなデバイスを富士通MB84256のような８ビット３２KのCMOSSRAMとしている。これによれば、スピードを５０ｎｓに落とすことができる。かなり高速のデバイスを用いればリターンを減少させることができる。その理由は、リアライザシステムのクロスバーチップ相互接続遅延が主な原因となり始めるからである。ダイナミックメモリデバイスを用いてはいない。その理由は、ダイナミックメモリデバイスではこれらを規則的にリフレッシュしなければならず、リアライザシステムに種々の問題が生じる。入力設計部にダイナミックメモリが必要な場合は、入力設計部はおそらくリフレッシュロジックを具えている。しかしながら、実現された設計部が１００％の設計スピードで動作できないので、設計部をリフレッシュさせることは成功しない。実際、デバッギングの際に、設計部の実行を停止させることが望ましい。すなわち、設計部はシステムの一部分であり、リフレッシュをするためには入力設計部に具わっていない他のいくつかの構成要素に依存しなければならない。つまり、設計部にスタティックメモリを必要とする場合、ダイナミック設計メモリのリフレッシュを行うことは非現実的である。スタティックメモリによればリフレッシュサイクルを無視できるため、ダイナミックメモリを設計部内に実現することができる。このようにして、スタティックデバイスを用いて設計部メモリを具体化する。

【００３２】１.３.１.２　ロジックチップをRAMとクロスバーとの相互接続に使用する

理想的には、単一のロジックチップを用いてＲＡＭとＸレベルクロスバーとを相互接続する。この際、すべてのＬ—Ｘ相互接続バスと同様に、すべてのＲＡＭ信号ピンを接続するのに十分なピンを用いる。実用的なリアライザシステムメモリモジュールでは、単一のロジックチップが実行困難な程多くのピンを必要としている。例えば８個の３２Ｋ、８ビットＲＡＭから成る２つのバンクを、１２８個のＬ—Ｘバスを有するモジュール中に用いるものと仮定する。各々のＲＡＭバンクは、１５個のアドレスピンと、８個の書込みイネーブルピンと、６４個のデータピンとを具えている。２個のバンク及びＬ—Ｘバスは、３０２個のピンと、ホストインタフェースバスのためのピンとを必要としている。これは、使用可能なロジックチップのピン数の２倍である。１より多くのロジックチップを用いなければならない。ここで述べたアーキテクチャでは、多くの小型ロジックチップを用いており、これらのチップにはアドレス、コントロール及びデータバスに関する特別の機能が与えられる。

【００３３】１.３.１.２.１　メモリアドレスロジックチップ

図２３において、アドレス及びコントロールロジックチップを、“ＭＡ０”及び“ＭＡ１”で示している。ＲＡＭをバンクに分割する。このバンクを各々のMAチップで制御する。モジュールによって実現すべき分離設計部メモリの最大数と同数のMAチップを設ける。その各々に、クロスバーとつながっているＬ—Ｘバスのセット、すなわち、バンクのアドレス及びコントロールラインにとって必要なだけのバスを設ける。ＭＡ０及びＭＡ１は別のバスの組を使用する。例えば、各々ＲＡＭの半分に接続されている２個のMAチップによって、２個の独立メモリを実現することができる。１個の大型メモリを実現する場合、両方のＬ—Ｘバスの組を用いて、アドレス及びコントロール回路網を両MAチップに接続する。各MAチップはバンク内の全ＲＡＭのアドレス入力を制御する。アドレス入力を、単一のバスで結びつける。各々のMAチップは、個々で、RAMへの制御入力を制御し、データをアドレス指定したＲＡＭにのみ書き込むことができるようにしている。つまり、各MAチップをアクセス可能とするために、ホストインタフェースバスに接続するとともに、このメモリモジュールのすべてのロジックチップと共通なコントロールバスに接続している。図２４は、いかにしてMAチップをＸレベルクロスバー及びＲＡＭチップに接続するかを、さらに詳細に示している。図に示すように、ロジック及びデータバスに従ってMAチップを構成する。すべてのアドレスがクロスバーからMAチップに入る。通常、（バスインタフェースを非活動状態とした場合）、ＲＡＭアドレスビットの数に相当するアドレスビットの部分をバスし、MAチップによって制御されるバンク中のＲＡＭをアドレス指定する。その他のアドレスビット及び設計部の書き込みイネーブルによって、各々のＲＡＭの書き込みイネーブル信号を制御するデコーダロジックを駆動する。この設計部メモリに必要とされる構成に応じて、ロジックを構成する。例えば、設計部メモリが１個のＲＡＭと同じビット幅を有し、設計部が書き込みイネーブルを主張する場合、アドレスビットに従い、ただ一つのＲＡＭ書込みイネーブルが主張される。設計部メモリが１個のチップの２倍の幅を有する場合、一対のＲＡＭ書込みイネーブルが主張される。メモリのデータバス幅のサブセットを各々制御している１より多くの書込みイネーブルを有する設計部メモリを望む場合、幾つかの設計書込みイネーブル回路網を用いることができる。各回路網は、MA及びMDチップ中のデコードロジックの構成を適切なものとし、上述のラインに沿って作動する。このことは、ＭＡチップへつながっているＬ—Ｘバスと、MDチップへつながっているコントロールバスバスとの使用可能度に依存している。バスインタフェースロジックによって、ホストインタフェースバスを介し、ホストはこのＲＡＭをアクセスさせることができる。この組となっているＲＡＭをバスを用いてアドレス指定する場合、バスインタフェースは、アドレスマルチプレクサ（‘ｍｕｘ’）を切り換え、ＲＡＭをそのアドレスにアドレス指定する。

ホストが1個のRAMに書込みを行う場合、バスインタフェースロジックは、信号をデコーダロジックに送信する。デコーダロジックは、アドレスビットを使用し、RAMを駆動せずに、適切なRAM書込みイネーブルを主張する。最終的には、MDチップ中のデータパスを制御するのに、幾つかの信号を必要とする。MDチップのすべてを、MDチップと同一のL—Xバスに接続してはいないので、MDチップは、設計部からのアドレス及びコントロール信号をアクセスする必要はない。コントロールバスをすべてのMA及びMDチップに接続し、これらの信号及びバスインタフェースコントロール信号をMDチップへ送信できるようにしている。

【0034】1.3.1.2.2　メモリデータパスロジックチップ

MDチップは、ビットスライス構成に従ってデータパスを操作する。クロスバーと交差してビットスライスを行うことによって、リアライザシステム中のマルチビットバスデータパスを相互接続する。チップ毎に1又は2ビットを用いて、バスはXチップと交差して広がっている。MDチップをビットスライスし、これらのバスとの接続を容易にしている。各MDチップをすべてのバンク中の各RAM中の同一のビットに接続するとともに、Xチップのサブセットに接続する。同一のRAMビットのすべてをMDチップ中で結びつけ、種々のビット幅及びサイズ設計部メモリの構成に柔軟性を持たせることができる。MDチップ中にロジック及びデータパスを適切に構成することによって、RAM幅の種々の倍数で、設計部メモリを構成する。‘n’個のMDチップ及び‘M’個のXチップを設ける場合、各MDチップをM／nの種々のXチップを用いて接続する。各データビットは、2個のL—Xバス、すなわち、クロスバー加算相互接続構成のために、分離I／O構成のためのDI及びDOバス又は共通I／O双方向性構成のための加算入力及び加算結果のいずれか一方である。このようにして、各MDチップは少なくとも2＊M／n個のL—Xバスを有している。これらに加えて、付加的なバスを設けることができる。これら付加的なバスを、MAのL—Xバスに重ねることができる。MDチップ、RAM及びRAMビット幅の数を選択し、これらの制約及び容量制約を適合させ、MDチップに用いられるロジックチップ中のピンの数を有効に用い、これを偶数となるようにしている。工業規格スタティックRAMチップは、双方向性データピン（DQと称する）を有する共通I／O構造を有しており、データイン及びデータアウトに用いられる。これは、アドレス入力ピン(ADDR）及び書込みイネーブルピン（WE）を有している。この実現例において、出力イネーブルピン及びチップ選択ピンは永続的にイネーブルされており、出力ピンは書込みイネーブルで制御する。必要な場合にはRAMの読出しを行い、アドレスデータをDQピンで駆動する。書込みイネーブルを主張する場合、データインをDQピンで受信する。この主張の終了時に、データをアドレスロケーションに書込む。規格デバイスは、書込みイネーブルの終了時にセットアップ中のデータのみを必要とし、また、ゼロ保持時間を必要とし、これによってデータバスの書込みイネーブル制御を可能としている。設計部メモリが共通I／Oを必要とする場合、設計部はトライステート回路網となる。これは、クロスバー加算構成を用いて実現される。すなわち、駆動ピンはそのイネーブルによって、別々にゲートされ、受信ピンを駆動する加算ORゲートに集められる。RAMDQデータピンを、図25に示されているようなMDチップ中に構成されるロジック及びデータパスによってインタフェースさせる。（一つのビットすなわちビット‘n’を図示する。他も同様である。）Lチップがトライステートドライバを有している場合に、Xチップ中の加算ゲートを駆動するイネーブルゲートを有しているように、Xチップ中の加算ゲートを駆動するイネーブルゲートを用いて各MDチップを構成する（MD‘n’を図示する）。設計部メモリ入力回路網によって、出力端子をイネーブルするとともに書込みをディゼーブルする場合、ロジックは、RAMの加算ゲートへの出力をゲートするとともに、受信ドライバをディゼーブルする。もし、そうでなければ、回路の値は加算ゲートからRAMへと伝達され、書込みイネーブルが主張されると書込みが可能となる。上述したように、設計部書込みイネーブル及び出力イネーブル信号がMAチップから（コントロールバスを介して）生じることに注意しなければならない。バスインタフェースロジックは図示していない。設計部メモリが分離I／Oを必要とする場合、これは、図26に示されているように、SRAMの共通I／Oから抽出される。出力イネーブルが主張される場合、データアウトは、常にSRAMのデータピンステートを反映している。書込みイネーブルが主張される場合、データインはSRAMのDQピンに伝達される。上記の図面では、設計部データビットに接続された1個のRAMのみを図示している。時には、数個のRAMを設けており、この場合、設計部メモリ中のロケーションの数は、単一のRAMチップの大きさの倍数となっている。このような場合、図27に示しているように、MDチップを構成する。幾つかのRAM各々のDQピンをこのMDチップに接続する。ローアドレスビットと設計部及びバスインタフェース制御信号とが、コントロールバスを介して、MAチップからMDチップへと伝達される。読出しの場合、アドレスのロービットは、マルチプレクサを介してRAMDQ出力のいずれか一つを選択する。選択された出力は設計部出力イネーブルによってゲートされ、前述の例と同様に設計部メモリデータアウトを構成する。設計部がその出力イネーブルを主張する場合、ドライバをイネーブルすることによって、データインはRAMDQ入力の内のいずれか一つに伝達される。

ローアドレスビット及び設計部書込みイネーブル信号によって駆動されるデコードロジックによって、駆動すべき適切なドライバを選択する。ＲＡＭチップの書込みイネーブルをＭＡチップによって駆動することを中止する。図２７は、分離Ｉ／Ｏ構成を示している。共通Ｉ／Ｏ構成は、クロスバー加算ゲートによって駆動されるデータインと、設計部出力イネーブル及び書込みイネーブルによってゲートされるデータアウトとに類似しており、図２５に示すように加算ゲート入力を駆動する。ホストインタフェースが、ホストインタフェースバスを介してこのメモリをアクセスする場合、ＭＡチップによって構成されるロジックは、バスをアクセスするための制御信号を出力する。この信号は、コントロールバスを介してＭＡから伝達される。バスが読出しを行う場合、バス読出しイネーブルは、マルチプレクサがアドレス指定したＲＡＭより選択したデータを、このＭＤチップに対応するホストインタフェースバスデータビットに伝達する。バスが書込みを行う場合、バスデータビットからのデータを他のマルチプレクサを用いてドライバにスイッチする。このデータは、通常の書込みと同じプロセスによって選択されたＲＡＭのＤＱピンへ伝達される。この説明は、単一の設計部メモリのデータバス幅から単一のデータビットを用いて構成した、ＭＤチップ構成を示していることに注意しなければならない。設計部メモリ構成によるものであり、且つモジュール中のＭＤ及びＲＡＭチップの数を必要とする場合、単にデータバスを適切に曲げることによって、１より多くのデータビットが各ＭＤチップ中に現れる。さらに、前記データバス及びコントロールラインを曲げることによって、一組の共通ＭＤチップを用い、１より多くの設計メモリを実現し、幾つかのメモリを具体化する。メモリモジュールにつながっているあるＬ—ＸバスをＭＡチップにのみ接続し、且つあるＬ—ＸバスをＭＤチップにのみ接続しているので、適切なＬ—Ｘバスを用いて設計部メモリに接続された回路網を相互接続するためだけに設計部変換相互接続プロセスを組立てる。

【００３５】１.３.１:３ 設計部メモリのための設計変換

オリジナル設計ファイル中で利用可能な構成のいずれか一つに対応する設計部メモリＲＡＭ基本要素を用い、入力設計部中の設計部メモリを特定する。設計部変換方法は一組の予め定義した部分的ネットリストファイルに基づくものである。この内の一つはメモリモジュールのロジックチップの各々のためのものであり、上で示したように、特別のメモリ構成をするために構成すべきすべてのロジック及びデータバスに関するステートメントを用いている。予め定義されたファイルは、相互接続を用いて、設計部メモリアドレス、データ及びコントロール接続を行うのに用いられるモジュールＩ／ＯピンのＩ／Ｏピン数仕様を除いて、完全なものである。この方法は以下のとおりである。：設計変換のセクションにて述べるように、以下に示すような設計部メモリに対して特別な例外があるものの、一般的な方法を設計変換に用いる：

・設計部リーダは、特定のベクトルメモリに対するメモリ基本要素を、設計データ構造に読出す。どの構成を用いるかを特定するためのデータを、メモリのデータ構造レコード中に記録する。

・変換ステージが、構成を利用可能であり、且つ、ピンが構成と正しく対応していることをチェックする。

・ユーザは、どのボード上のどのＬチップ位置にメモリモジュールが搭載されているかをパーティショナ（partitioner）に告げる。このデータに基づき、パーティショナは、一般的分割化アルゴリズムに従って、記憶のためのメモリモジュールを選択する。択一的に、ユーザは、このデータをオリジナル設計ファイル中の基本要素と関連づけることによって、メモリを特定のモジュールに割り当てることができる。設計部リーダは、メモリの基本要素レコード中に、オリジナル設計ファイルを具えている。

・次に、インタコネクタは、メモリに接続された回路網及びピンを、特定のＬ—Ｘ相互接続バスに割当てる。アドレス及びコントロール回路網をＭＡチップに接続している特定のバスにのみ割当て、且つ、データ回路網をＭＤチップと接続しているバスにのみ割当てることができるという制約を条件として、インタコネクタはバスの割当てを行う。各クロスバーチップセットの回路網相互接続能力を決定する場合、これらのセットを拒否する場合、及び必要とされるＭＡ又はＭＤチップを接続していないバスを得られない、又は使用することができない場合、相互接続を行う際に、これらの制約を適用する。

・リアライザシステム中の各ロジックチップに関するネットリストファイルに書込みを行う場合、各々の設計部メモリ回路網接続は：

１）ＭＡ又はＭＤのいずれかを相互接続手続によって、基本要素が選択するバスに接続するかを決定すること

２）通常のロジックチップＩ／Ｏピン数を得る場合に説明したのと同様の手続を用い、バス数とＭＡ／ＭＤチップ数とからロジックチップＩ／Ｏピン数を得ること

３）これまで他の回路網に割当てられていないこのＭＡ／ＭＤチップの回路網にからの、予め定義されたアドレス、データ又は制御接続を選択すること

４）ステートメントを、このロジックチップのネットリストファイルに加え、このロジックチップＩ／Ｏピン数を、予め定義した設計部メモリ接続に接続するのに用いることを明示することによって、ネットリストされる。

・ネットリストファイルを、ネットリスト変換ツールを用いて、構成ビットパターンに処理するとともに、Ｌチップ及びＸチップのネットリストファイルとしてのロジックチップにロードする。

【００３６】１.３.１.４ 具体的なメモリモジュール

設計

図２８は、好適例において用いられる、メモリモジュールの設計部を示す図である。これを、図２３に示した上述の説明に基づく構成に従って、アーキテクトすることに注意しなければならない。XC3090　ＬＣＡロジックチップに代わるＬチップソケットに、プラグを差し込み接続するように構成する。このようにして１２８個のＬ－Ｘバス、すなわち、各々３２個のＸチップにつながっている４個のバスを設ける。共通Ｉ／Ｏを有する３２Ｋ、８ビットスタティックＲＡＭチップを、８個のＲＡＭの各々の２個のバンク中に用いる。各バンクは、それ自体のＭＡチップ、XC2018　ＬＣＡを有している。各ＭＡチップは、８個のアドレスバス及び８個の書込みイネーブルを用いて、そのＲＡＭを制御する。各ＭＡチップを、モジュール中のすべてのＭＡ及びＭＤチップが共有している制御バスに接続するとともに、ホストインタフェースバスに接続する。残りのピンは、クロスバーと接続している。各々異なるＸチップとつながっている、２８個のＬ－Ｘバスを設ける。ＭＡチップ０は、一組のバス、バス０を使用し、ＭＡ１はバス１を使用する。これによって、２個の独立設計ＲＡＭに対する別々のアドレス及びコントロール回路網が与えられる。完全な３２個のＬ－Ｘバスよりも少ないバスを接続する。これは、単に、XC2018のピンの数が制限されているからに他ならない。設計変換の間、このモジュールにおけるミッシングバスに対応する、相互接続Ｌ－Ｘバステーブルにおけるバス構成要素が利用できないことに注意しなければならない。このため、回路網を、バス構成要素を介して相互接続できない。８個のＭＤチップには、すべてXC2018　ＬＣＡを使用する。３２個のＸチップを設ける場合、（上述の方法によれば）各々のＮＤチップは３２／８＝４個の異なるＸチップを接続している。各チップは、設計部メモリデータビットに用いられる２＊Ｍ／ｎ＝８個のバスを有している。その内の２個は、各Ｘチップにつながっている。各Ｘチップにつながっている付加的な２個のバスを設け、以下に示すように、モジュールを、１２８ビットベクトルメモリとして使用できるようにする。好適例において実現されるホストインタフェースバスを、Ｒバスと称する。Ｒバスは、すべてのＬチップポジションを、付加的なピンを用いて接続する。これについては、ホストインタフェースのセクションで説明する。５個の異なる設計部メモリ構成を、このモジュール中で用いることができる。以下のチャート及び図２８において“バス０”は、各Ｘチップからつながっている一組のＬ－Ｘバスを示しており、“バス１”は他の一組を示している。

・８ビット５１２Ｋの１個のメモリ：Ｌ－Ｘバス０及び１を介した（ＭＡ０及びＭＡ１の両方に接続できるように二重にしている）１９個のアドレス及び２個のコントロール（ＷＥ、ＯＥ）、Ｌ－Ｘバス２及び３を介した１６個のデータ（ＤＩ／ＤＯ又はドライバ／レシーバ）。各ＭＤチップは、１６個のＲＡＭに接続された１個のデータビットを有している。

・１６ビット２５６Ｋの１個のメモリ：Ｌ－Ｘバス０及び１を介した１８個のアドレス及び２個のコントロール、Ｌ－Ｘバス２及び３を介した３２個のデータ。各ＭＤチップは、各々８個のＲＡＭに接続されている２個のデータビットを具えている。

・３２ビット１２８Ｋの１個のメモリ：Ｌ－Ｘバス０及び１を介した１７個のアドレス及び２個のコントロール、Ｌ－Ｘバス２及び３を介した６４個のデータ。各ＭＤチップは、各々４個のＲＡＭに接続されている４個のデータビットを有している。

・８ビット２５６Ｋの２個のメモリ：各々、Ｌ－Ｘバスを介した１８個のアドレスと、２個のコントロールとを有している。バス０は一方のメモリ（ＭＡ０）のためのものであり、バス１は他方のメモリ（ＭＡ１）のためのものである。各々は、バス２及び３を介した１６個のデータを有している。各ＭＤチップは、８個のＲＡＭに接続された、各々のメモリのための１個のデータビットを有している。

・１６ビット１２８Ｋの２個のメモリ：各々は、Ｌ－Ｘバスを介した１７個のアドレスと２個のコントロールとを有している。バス０は一方のメモリのためのものであり、バス１は他方のメモリのためのものである。各々、バス２及び３を介した３２個のデータを有している。各ＭＤチップは、４個のＲＡＭに接続された各メモリのための２個のデータビットを有している。コントロールバスは、一般的にすべてのＭＡ及びＭＤチップに接続された１２個のバスから成っている。１２個のバスは最大コントロール構成を保持する必要がある。この構成は三つのアドレスビットである。すなわち、設計書込みイネーブルと、２個の２５６Ｋ、８ビット設計部メモリの各々のための設計部出力イネーブル信号とに、バス書込みイネーブル及びバス読出しイネーブルを加えたものである。

【００３７】１.３.２　刺激及び応答

リアライザシステムを多数使用することは、ホストコンピュータの刺激信号送信と、設計部への応答信号及び設計部からの応答信号の捕捉とに依存している。このことを、バッチ形式で行う場合、すなわち、信号の大部分を一度に送信及び収集する場合に、ベクトルメモリを用いる。このことを、一回に一つの信号で行う場合には、スティミュレータ及びサンプラを用いる。

【００３８】１.３.２.１　刺激を与えるためのベクトルメモリ

連続的且つ反復的な刺激のストリームを、シミュレーション適用のような、テストベクトルの高スピード反復適用のために実現される設計部中の一組の回路網に供給することが時々必要となる。このことは、実現される設計

部の回路網にメモリをインタフェースさせること、刺激ベクトルをホストコンピュータからメモリに書込むこと、更には、順次にメモリを１回ないし数回読出し、刺激を設計部に送ることによって行われる。連続的且つ、リニアなメモリロケーションを読出す必要があるため、アドレスストリームを２進カウンタによって設ける。図２９はこのような刺激ベクトルメモリを達成するための手段を示している。規則的なクロック信号ＥＣＬＫはプロセスを制御する。ＥＣＬＫを周期化、すなわち、各刺激ベクトルの度毎にハイとローとを発生させる。２進カウンタはアドレスシーケンスを提供する。ＥＣＬＫがハイになると、カウンタは次の刺激ベクトルのアドレスまでカウントアップする。次の刺激ベクトルのアドレスは、ＥＣＬＫの周期の間ＲＡＭによって読出される。ＥＣＬＫが次にハイになると、ちょうど読出された刺激ベクトルの値がＤフリップフロップのクロックとなる。フリップフロップの出力信号は刺激ベクトルの値で刺激される回路を駆動する。フリップフロップは、ベクトル間に必要なクリーントランジッションを与える。その理由は、ＲＡＭ出力が正しい値に安定する以前に、その読出サイクルの間変動し得るからである。このプロセスは繰り返され、一連の刺激ベクトルが実現される設計部に与えられる。この構造は繰り返され、刺激が多くの回路網に提供される。刺激ベクトルをＲＡＭに書込むのに用いられるホストコンピュータへのインタフェースは、簡単のため図示していないが以下に引用する図面により更に詳細に示す。

【００３９】１.３.２.２　応答捕捉のためのベクトルメモリ

同様に、実現される設計部からの応答を捕捉するーモードでは、連続的なサンプルのストリームすなわち一組の回路網からのベクトルを捕捉する。この時、ロジックアナライザが現実のハードウエアデバイスから捕捉を行う。このことは、メモリを、実現される設計部の回路網にインタフェースさせ、実現される設計部が順次に動作するときに回路網からのベクトルをメモリに書込み、更に捕捉された応答ベクトルを、解析のためにホストコンピュータへ戻すことによって行われる。連続的且つ、リニアな一連のメモリロケーションを読出す必要があるため、前記と同様、アドレスストリームを２進カウンタによって設ける。図３０はこのような応答ベクトルメモリを開発する手段を示している。刺激メカニズムのように、クロック信号ＥＣＬＫがプロセスを制御する。各応答ベクトルの度毎に、ＥＣＬＫの同期をとる。２進カウンタはアドレスシーケンスを提供する。ＥＣＬＫがハイになると、カウンタは次のベクトルのアドレスまでカウントアップする。ＥＣＬＫがローになると、応答ベクトルの値がトライステートドライバによってＲＡＭＤＱデータピンに伝達され、書込みのためにＲＡＭがイネーブルされる。ＥＣＬＫが再びハイになるとこの値はＲＡＭロケーションに書込まれ、ＲＡＭ書込みイネーブル及びトライステートドライバイネーブルはディゼーブルされ、カウンタは次のベクトルのアドレスまで進む。このプロセスは繰り返され、実現される設計部からの一連の応答ベクトルを記録する。この構造は繰り返され、刺激が多くの回路網に供給される。刺激ベクトルをＲＡＭに書込むために用いられるホストコンピュータへのインタフェースは、簡単のため図示していないが、以下で引用する図面において更に詳細に説明する。一般的に、実現される設計部を刺激し、これらの応答を発生させる。刺激が刺激ベクトルメモリから生じる場合、両ベクトルメモリは同一のＥＣＬＫ信号を用いている。ＥＣＬＫ信号は、新しいアドレスがカウンタから読み取られ、ＲＡＭをアドレス指定するとともに、データが読出され、刺激Ｄフリップフロップをセットアップするのに十分長くハイである必要がある。また、ＥＣＬＫ信号は、刺激が実現される設計部に影響を及ぼし、この影響に対するすべての応答が安定し、且つ、これらの応答がＲＡＭに書込まれるのに十分長くローでなければならない。刺激がいずれかから生じる場合、応答回路網を正しくサンプリングするために、応答ベクトルメモリのＥＣＬＫ信号は実現される設計部と同期されている必要がある。

【００４０】１.３.２.３　刺激及び応答のためのベクトルメモリ

図３１で示されているように、刺激及び応答ベクトルメモリシステムに関して上記のように定義された、刺激及び応答ベクトルメモリの機能を組合わせることができる。ＲＡＭビットは、たとえ同一のＲＡＭデバイスであっても、刺激又は応答のいずれか一方に自由に割当てることができる。その理由は、ＥＣＬＫがハイのときに刺激読出し機能が生じ、そして、ＥＣＬＫがローのときに応答書込み機能がこれに続くからである。トライステート応答ドライバを両方とも刺激Ｄフリップフロップ入力とし、同一のＲＡＭＤＱデータピンに接続することによって、一つのビットを刺激及び読出しの両方に用いることができる。シンプル刺激ベクトルメモリと組合せ刺激／応答ベクトルメモリとの重要な相違点は、刺激ベクトルを、１回だけＲＡＭから読出すことができるということである。その理由は、ＲＡＭビットを刺激のみに用いた場合でさえ、各メモリロケーションをＥＣＬＫの半周期のローの時に書込むからである。このことは、ＲＡＭチップのすべてのビットを刺激に用い、且つＥＣＬＫが書込みイネーブルを主張しない場合にのみ避けることができる。前の図面は、一般的な方法でベクトルメモリを実現したものを図示している。更に、点線は、いかにしてロジックチップ（“MAチップ”及び“MD‘n’”）を構成することでベクトルメモリロジック機能を実現することができるかを示すものである。これらロジックチップは、適切にＲＡＭチップ及びリアライザ相互接続（Ｘチップ）に接続されている。ベクトルメモ

リと、ソフトウエアからの刺激を電気的な型に再び戻す変換については、米国特許第４，７４４，０８４号明細書において詳細に説明されている。この内容を、参考のためにここで用いる。

【００４１】１.３.２.４　フォールトシミュレーションのためのベクトルメモリ

リアライザフォールトシミュレーションシステムについては、これについてのセクションにおいて説明する。フォールトシミュレーションでは、応答はベクトルメモリに捕捉されず、その代わりに、フォールト応答ベクトルメモリによって所定の良好な回路の応答と比較される。フォールト応答ベクトルメモリは、以下の点において上で示した簡易刺激ベクトルメモリと同一のものである。すなわち、MDチップのフリップフロップの出力を用いて回路網を駆動する代わりに、出力はＸＯＲゲートによって回路網の値と比較される。ＸＯＲゲートを、ＥＣＬＫが同期をとるセットフリップフロップに接続し、回路網とメモリとの差を表示しているＸＯＲゲートがハイの場合フリップフロップをセットする。ホストは、ホストインタフェースを介してこのセットフリップフロップを読出すことができ、差が検出されているかどうかを調べることができる。

【００４２】１.３.２.５　実現される設計部におけるベクトルメモリの相互接続

実現される設計部へのベクトルメモリの接続方法は多くの方法が考えられる。１以上のロジックチップに直接接続され、及び／又は相互接続バスのいずれか又はすべてに接続されたベクトルメモリを用いて、リアライザシステムを設計することができる。例えば、ベクトルメモリを、Ｌチップ及びＸチップを用いてロジックボードに取り付けることができるとともに、ボードとは離れているＸ—Ｙバスに接続することができる。ベクトルメモリを、Ｙレベルクロスバーの Ｙチップボードに取り付けるとともに、Ｘ—Ｙ及びＹ—Ｚバスに接続することもできる。ベクトルメモリを、ロジックチップの代わりにＬチップロケーションに取り付け、Ｌチップロケーションに作用するＬ—Ｘバスに接続するというテクニックもある。この場合、これらＬ—Ｘバスをベクトルメモリと Ｘチップとの間にのみ接続する。Ｘチップを構成することによって、実現される設計部の回路網への接続を行い、ベクトルメモリを回路網に接続する。この際、回路網はＸレベル相互接続を介してつながっている。モジュールの方法でロジックチップをベクトルメモリモジュールに置き換え、リアライザシステムを、必要な数の又は必要よりも少数のベクトルメモリを用いて構成することができる。リアライザ設計部メモリを、Ｌチップロケーション中の１以上のロジックチップに代えて取り付けているため、このテクニックを用いて、共通ハードウエアメモリモジュールを設計部メモリモジュール又はベクトルメモリモジュールとして用いることができる。メモリモジュール中にロジックチップを構成するとともに、リアライザシステムの相互接続を適切に行うことによって機能を選択する。これは、好適例において用いられているベクトルメモリアーキテクチャである。

【００４３】１.３.２.６　特別なベクトルメモリ設計部

好適例において、共通メモリモジュールを、設計部メモリ及びベクトルメモリ応用の両方のために使用する。その一般的なアーキテクチャ及び設計は、設計部メモリのセクションにおいて説明し、ここでは説明しない。いかにして、モジュールをベクトルメモリとして用いるかの詳細については、以下に示すとおりである。以下の２個の図面は、ホストインタフェースからの完全読出し／書込みアクセスを用いて、組合せ刺激／応答ベクトルメモリのためのMA及びMDチップ中に前記と同様のロジックを構成することを示している。ホストコンピュータが非活動状態である場合、すべての動作は上記簡単な例にて示したのと同一のテクニックに従っている。図３２において、ホストインタフェースを介してホストが出力するＥＣＬＫ信号を、相互接続を介してMAチップに相互接続している。ＥＣＬＫ信号は、各MAチップで構成されるアドレスカウンタの同期をとる。各々、一組のＲＡＭを制御している１以上のMAチップをモジュール中に設けているので、各MAチップは、ベクトルアドレスカウンタのコピーを有している。すべてのカウンタは、同一のコントロール（ＥＣＬＫ及びバスインタフェースからのリセット信号）を得ているため、その各々は常に他のカウンタと同一のアドレスを送信する。通常（バスインタフェースが非活動状態の場合）、アドレスがカウンタ出力から送られ、ＲＡＭのアドレス指定を行う。ＥＣＬＫがロー状態（書込み応答位相）の場合、デコーダロジックは、前述の例と同様にすべてのＲＡＭ書込みイネーブルを主張する。ＥＣＬＫは、コントロールバスにも伝達され、MDチップのロジックを駆動する。MDロジックは刺激及び応答ベクトル値それ自体を処理する（図３３）。通常（バスインタフェースが非活動状態の場合）、ＥＣＬＫがハイ状態のとき、ＲＡＭは刺激ベクトル値を読出し、ＥＣＬＫがロー状態になると、ＲＡＭとフリップフロップとを同期させる。フリップフロップは、上記と同様に各回路網に刺激を与えるためのものである（その内の一つを図示する）。従って、刺激を相互接続Ｘチップを介して回路網へ伝達する。ＥＣＬＫがロー状態の場合、すべてのトライステートイネーブル（ｅ０、ｅ１、・・・ｅｎ）が主張され、相互接続（２個を図示する）を介して回路網から出力される応答値をマルチプレクサを介してＲＡＭＤＱデータピンに伝達する。ホストコンピュータが、ホストインタフェースバス（特に、好適例のＲバス）を介してこのメモリをアクセスする場合、各々のMAチップ中に構成されるバスインタフェースロジックが活動状態となる。これは、アドレスマ

ルチプレクサ（ｍｕｘ）を切り換え、バスがＲＡＭのアドレス指定を行う。バスサイクルがＲＡＭに書込みを行うためのものである場合、デコーダロジックは、アドレスビットを用いてどのＲＡＭに書込みを行うべきであるかを解読するとともに、適切な書込みイネーブル信号を出力する。ＲＡＭを選択するのに必要とされるアドレスビット及び読出し及び書込み制御信号も、コントロールバスを介してＭＤチップに伝達される。ＭＤチップにおいては、バスが読出しサイクルを行う場合、デコーダロジックはすべてのトライステートＲＡＭＤＱピンドライバをディゼーブルし、アドレスビットを用いて読出しマルチプレクサを介してアドレス指定されたＲＡＭのＤＱデータ出力を選択し、更には、バス読出しイネーブル信号がデータ値をこのビットのためのホストインタフェースバスのデータラインに伝達する。バス書込みサイクルにおいて、デコードロジックは、書込みマルチプレクサを用いて、応答を与える回路網ではなく、ホストインタフェースバスのデータラインから生じるデータ値を選択するとともに、アドレス指定されたＲＡＭのためのトライステートＲＡＭＤＱドライバをイネーブルし、データをＲＡＭ入力へ伝達する。

【００４４】１.３.２.７　ベクトルメモリの設計変換及び仕様

回路網をベクトルメモリに接続すべきであるということを説明するために、ユーザは、回路網に入力設計に関する特別な特徴を付加し、特定のベクトルメモリ及び接続が刺激のためのものであるのか又は応答のためのものであるのかを説明する。設計部変換方法は、一組の所定の部分的ネットリストファイルに基づくものであり、この内の一つは各モジュールのロジックチップのためのものであり、前記と同様ベクトルメモリ刺激及び応答接続と、ベクトルメモリデータバス及びコントロールロジックと、バスインタフェースロジックとに関するステートメントを用いている。この方法では、ＥＲＣＧＡネットリスト変換ツールは、任意の出力端子又はＩ／Ｏピンに接続されていない入力端子、及び任意の入力端子又はＩ／Ｏピンに接続されていない出力端子のような、通常は接続されていないネットリストファイル中の基本要素及び回路網のためのロジック及び相互接続を構成することはない。各ベクトルメモリビットに対する刺激接続及び応答接続のためにロジックを設ける。ネットリストに供給されるいずれか一方の相互接続のみが実際に構成され、他方は構成されない。その理由は、通常それをネットリストに接続しないからである。予め定義されたファイルは、相互接続を用いてベクトルメモリ刺激接続とベクトルメモリ応答接続とを接続するのに用いるモジュールＩ／ＯピンのＩ／Ｏピン数の仕様を除いて、完全なものである。各ファイルにおける刺激及び応答接続の数を、何個のＩ／Ｏピンをファイルのロジックチップ中に用いることができるか、どの程度のロジックを各チップに、更には全体としてどの程度のロジックをモジュールに設けることができるか、によって決定する。その方法は、以下のとおりである。：設計部変換のセクションにおいて説明したように、以下のようなベクトルメモリの特別な例外を有するものの、一般的な方法を設計部変換に用いる：

・設計部リーダは、ベクトルメモリ接続のために設けられた回路網を識別するために入力設計ファイルからの特性情報を読出し、且つ、バスインタフェースロジックではなく、回路網に接続された１以上のベクトルメモリ基本要素を、その設計部データ構造に組込む。設計部リーダは、ホストインタフェースクロック発生器及びベクトルメモリ基本要素に接続されたＥＣＬＫ回路網を作り出す。

・パーティショナとは、ユーザが、メモリモジュールを取り付けるボード上のいずれかのチップ指定するということである。このデータに基づき、パーティショナは、ベクトルメモリ基本要素を通常の方法でメモリモジュール中に分割する。

・インタコネクタは、他のロジックチップ基本要素と同一のベクトルメモリ基本要素を処理し、これらを回路網中の他の基本要素を用いて接続しているＬ—Ｘバスを決定する。

・リアライザシステム中の各ロジックチップのネットリストファイルに書込みを行う場合、各ベクトルメモリ回路網接続は以下によってネットリストされる：

１）どのロジックチップが、相互接続手続によって基本要素が選択したバスを接続するかを決定する。

２）通常のロジックチップＩ／ＯピンナンバをＩ得る際に説明したのと同様の手続を用いて、バスナンバ及びロジックチップナンバからロジックチップＩ／ＯピンナンバをＩ得る。

３）これまで他の回路網に割当てられていないロジックチップに関する回路網から、予め定義された刺激又は応答ベクトルメモリ接続を選択する。

４）ステートメントをこのロジックチップのネットリストファイルに加え、このロジックチップＩ／ＯピンナンバをＩ、予め定義されたベクトルメモリ接続に接続するために用いることを明示する。

・設計変換システムは、対応テーブルファイルも送出し、回路網の名称をベクトルメモリ及びベクトルメモリビット位置と関連づけ、動作中使用する。

・ＥＲＣＧＡネットリスト変換ツールは、用いられるベクトルメモリ刺激及び応答入力端子のロジック及び相互接続のみを構成する。

【００４５】１.３.２.８　スティミュレータ

スティミュレータは、単一の記憶ビットとし、ホストコンピュータで制御し、設計部の回路網を駆動する。スティミュレータは、ホストが入力信号を設計部に供給するのに用いられる。２種類のスティミュレータ、すなわ

ち、ランダムアクセスタイプとエッジ検知タイプとを設ける。実際のランダムアクセススティミュレータは、フリップフロップであり、その出力信号はホストインタフェースバスを介し、ホストが必要に応じてデータをロードする設計部回路網を駆動する。ランダムアクセススティミュレータは、設計部の動作を変化させることなく、他の刺激された回路網に呼応して、常に値を変化させることのできる回路網を刺激するのに用いられる。このような回路網の一例としてはレジスタへのデータ入力がある。各スティミュレータは唯一のバスアドレスを有し、ホストがデータをこのアドレスに書込む場合に、バスインタフェースロジックは、データをＤ入力に与えるとともにスティミュレータフリップフロップのクロック入力の同期をとる（図３４）。エッジ検知タイプのスティミュレータは、設計部の動作、例えばレジスタへのクロック入力を修正するための他の回路網と同期しながら変化しなければならない回路網を刺激するのに用いられる。第２フリップフロップを、ランダムアクセススティミュレータと設計部回路網との間に配置する。同期をとらなければならない一群のこのようなスティミュレータのすべてを共通クロックに接続する。新しい一組の回路値を入力するために、ホストは、新しい値を、たとえどのようなオーダであっても、上記と同様に、ホストインタフェースバスを介して各スティミュレータの第１フリップフロップにロードする。新しい値が設計部にすべて供給される必要がある場合、ホストは、共通‘同期クロック’を周期化し、一度にすべての値を第２フリップフロップにロードし、このようにしてすべての回路網を同時に駆動する（図３５）。

【００４６】１．３．２．９　サンプラ

サンプラは、単一の記憶ビットであり、ホストコンピュータによって制御され、設計部の回路網を受信する。サンプラはホストによって使用され、設計部からの出力信号を捕捉する。サンプラの最も簡単な形は、Ｄ入力端子で設計部回路網を受信し、同期をとることができ、且つ、ホストインタフェースバス及びバスインタフェースロジックを介してホストが必要に応じて読出すことのできるフリップフロップである。通常、多数のサンプラを、共通‘サンプルクロック’に接続する。サンプラデータ出力は、‘サンプルクロック’出力と同様に、唯一のバスアドレスを有している。ホストは、クロックを周期化し、一群のサンプルを取り出し、その後、サンプリングされたデータ値を一つ一つ読出す（図３６）。必要とされるホストＩ／Ｏの数を削減するために、第２フリップフロップを付加的に加え、変化検出サンプラを構成する。第２フリップフロップを、サンプリングフリップフロップと同一のクロックに接続し、その入力端子をサンプラの出力端子に接続する。結果的に、第２フリップフロップは最も新しいクロック周期前にサンプラが有していた値を保持している。２個のフリップフロップ出力をＸＯＲゲートで比較する。ＸＯＲゲートは、サンプリングされた値の変化のため２個のフリップフロップが相違する場合にハイ状態の値を出力する。一群のサンプラからの全ＸＯＲ出力信号をホストが読出し可能なＯＲゲートによって加算する。上述したように、‘サンプルクロック’を周期化することによって、回路網をサンプリングした後、ホストは、まず第１にこのＯＲゲートの‘変化’値をチェックし、グループ中のどの値が変化したのかを調べる。変化していない場合には、これらサンプラのいかなる値をも読出す必要がない（図３７）。

【００４７】１．３．２．１０　スティミュレータ及びサンプラの設計変換及び仕様

サンプラ及びスティミュレータフリップフロップ、ロジックゲート及びバスインタフェースロジックを、リアライザシステムロジックチップ中に実現する。回路網を、サンプラ又はスティミュレータに接続すべきであることを説明するために、ユーザは、回路網に、入力設計に関しての特別な特性を与え、スティミュレータ又はサンプラの特定のタイプとグループの同一性とを識別する。スティミュレータ及びサンプラを構成し、これらを、設計部の残りの部分及びバスインタフェースに接続するために、設計変換ソフトウエアシステムを用いる一般的な方法は以下に示すとおりである：設計変換のセクションにおいて説明したように、以下のようなスティミュレータ及びサンプラに関しての特別な例外があるものの、一般的な方法を設計部変換に用いる：

・設計部リーダは、特性情報を入力設計ファイルから読出し、スティミュレータ及び／又はサンプラのために設けられた回路網を識別し、バスインタフェースロジックではなく、回路網に接続されたスティミュレータ及びサンプラの基本要素を設計データ構造に組み込む。

・システムパーティショナは、このような基本要素の各々が、ロジックチップ中に何個の等価ゲートを有しているかについてのデータベースを有している。システムパーティショナは、バスインタフェースロジックの等価ゲート指数も有している。このデータに基づき、システムパーティショナは、その通常の分割化アルゴリズムに従ってスティミュレータ及びサンプラをロジックチップに割当てる。この際、システムパーティショナが、バスインタフェースロジックのサイズによって、ロジック容量の限界を小さくするという付加的な条件を課し、１以上のスティミュレータ及び／又はサンプラを有するロジックチップの各々が、バスインタフェースロジックブロックを有しなければならないということを説明する。

・インタコネクタは、他の基本要素と同じく、スティミュレータ及びサンプラ基本要素を処理する。

・リアライザシステムにおける各ロジックチップのネットリストファイルに書込みを行う場合、以下の手続を用いて、各サンプラ又はスティミュレータ基本要素をネットリストする。

１）サンプラ又はスティミュレータを構成するゲート及び／又はフリップフロップの基本的ステートメントを、ステートメント分割化のためのロジックチップのネットリストファイルへ送信する。相互接続基本要素について説明したのと同様の方法に従って、サンプリング又は刺激される回路網に亘る付加的な回路網のネームを、サンプリング又は刺激される回路網のネームから得る。

２）これが、この特別なロジックチップファイルにネットリストされる第１スティミュレータ及びサンプラである場合、バスインタフェースの予め定義されたネットリストファイルセグメントを用い、バスインタフェースを構成する基本要素及び回路網をロジックチップに供給する。インタフェース毎に１回のみ使用されるバスインタフェース相互接続には前記ファイルセグメントで定義される標準的なネームが与えられる。スティミュレータ又はサンプラロジックに接続されるものには、捕捉した回路網のネームが与えられる。このネームは、ステップ１において、基本要素を出力する際に用いたネームと整合している。

簡単ではあるが、一般的ではない方法を用いて、メモリモジュール又はユーザ指定のデバイスモジュールのロジックチップ中にのみ、スティミュレータ及びサンプラを実現する。このことは、ＥＲＣＧＡネットリスト変換ツールがどの出力端子又はＩ／Ｏピンにも接続されていない入力端子、及びどの入力端子又はＩ／Ｏピンにも接続されていない出力端子のような、通常接続されていないネットリストファイル中の基本要素及び回路網のためのロジック及び相互接続を構成しないということを仮定している。このことは、一組の予め定義された部分的ネットリストファイルに基づくものである。このファイルの一つは、各モジュールのロジックチップのためのものである。この際、以下のステートメントを用いている。

１）すべて共通‘同期クロック’に接続されている多数のエッジ検知タイプのスティミュレータ。

２）すべて同一の‘サンプルクロック’に接続された多数の変化検出サンプラ。

３）上記のすべてのためのバスインタフェースロジック。

予め定義されたファイルは、サンプラとスティミュレータとを相互接続を用いて接続するのに用いられるモジュールＩ／ＯピンのＩ／Ｏピンナンバの仕様を除き、完全なものである。コントロールバスを用いて、同期及びサンプルクロックのような共通信号をロジックチップ間に分配する。各ファイル中のスティミュレータ及びサンプラの数を、何個のＩ／Ｏピンがファイルのロジックチップ中に利用できるか、どの程度のロジックを各チップが有することできるか、及び全体としてのモジュールによって決定する。その方法は、以下に示すとおりである：

設計変換のセクションにおいて説明したような、一般的な方法を設計変換に用いる。この際、スティミュレータ及びサンプラに関して、以下のような例外がある。：

・設計部リーダは、スティミュレータ及びサンプラのために設けられた回路網を識別するために、入力設計ファイルからの特性情報を読出し、且つ、バスインタフェースロジックではなく、回路網に接続されたスティミュレータ及びサンプラ基本要素を、その設計部のデータ構造に組込む。

・パーティショナとは、ユーザが、メモリモジュール及びユーザ指定のデバイスモジュールを取り付けるボード上のいずれかのＬチップを指定するということである。このデータに基づき、パーティショナはまずメモリ及びＵＳＤ基本要素をモジュールに割当て、その後、その通常の分割化アルゴリズムに従い、各モジュール単位で利用することのできる数の限界に至るまで、スティミュレータ及びサンプラ基本要素をこのようなモジュールの残りへと分割化する。

・インタコネクタは、他のロジックチップ基本要素と同一なスティミュレータ及びサンプラを処理し、これらを、回路網で他の基本要素を用いて接続しているＬ－Ｘバスを決定する。

・リアライザシステム中の各ロジックチップのネットリストファイルに書込みを行う場合、各サンプラ又はスティミュレータ基本要素は以下によってネットリストされる：

１）どのロジックチップが、相互接続手続によって基本要素が選択したバスを接続するかを決定する。

２）通常のロジックチップＩ／Ｏピンナンバを得る際に説明したのと同様の手続を用いて、バスナンバ及びロジックチップナンバからロジックチップＩ／Ｏピンナンバを得る。

３）これまで他の回路網に割当てられていないロジックチップに関する回路網から、予め定義されたスティミュレータ／サンプラを選択する。

４）ステートメントをこのロジックチップのネットリストファイルに加え、このロジックチップＩ／Ｏピンナンバを予め定義されたサンプラ／スティミュレータに接続するために用いることを明示する。

・ＥＲＣＧＡネットリスト変換ツールは、使用するスティミュレータ、サンプラ及び関連あるバスインタフェースロジックのためのロジック及び相互接続を構成する。

両方の方法において、設計部変換システムは対応テーブルファイルも出力し、動作中に使用するために、回路網ネームを特定のスティミュレータ及びサンプラに関連させるとともに、アドレスをホストインタフェースバスで通信する。

【００４８】1.3.3　ユーザ指定デバイス

構成されたロジック及び相互接続チップの形態で実際に動作するハードウエア中に入力設計部を実現するために、他の実際のハードウエアデバイスをリアライザシステムに接続することが実用的であり且つ望ましい。マイ

クロプロセッサ又は他のＶＬＳＩ　ＩＣチップ、デジタル／アナログコンバータ、ディスプレイデバイス、入力キーボード及びスイッチ、記憶デバイス、コンピュータ入力／出力バス等のデジタル入出力を具える任意のデバイスを設けることができる。これらを、回路ボード又は大規模スケール構成素子のような、実現される設計部の一部を構成するデジタルシステムの一部とすることができる。これらのデバイスは、リアライザシステムのロジックゲート、フリップフロップ及びメモリ中で具現化することのできない、実現されるべき入力設計部の一部を示している。これは、ディスプレイのような物理的な理由によるもので、大規模記憶デバイスのような、リアライザシステムのリソースが不足しているため、又は標準的マイクロプロセッサのように、ロジック的な記述を利用することができないためである。代わりに、これらのデバイスは、すでに構成され正しいことが証明されている半通例のゲートアレイチップのような、ユーザがリアライザシステムリソースを用いて実現することを望まないデバイスともなり得る。その理由は、これを実現するために、リアライザシステムリソースを用いる必要がないため、又はユーザが設計部の実現される部分のデバイスを用いての正確な動作を試験したいためである。これらのデバイスは、リアライザシステムの一部ではないが、ユーザの設計の必要に応じて、ユーザによって指定されるものであるため、これらのデバイスを“ユーザ指定デバイス”（ＵＳＤ）と称する。ユーザがこのようなデバイスをリアライザシステムハードウェアに接続するのに用いるような標準的な手段を、リアライザシステムに設けるのに役立つような、多様なＵＳＤを設ける。この手段はユーザ指定のデバイスモジュール（ＵＳＤＭ）である。

【００４９】１.３.３.１　ユーザ指定のデバイスモジュール

ユーザ指定のデバイスモジュールは：

１）物理的にユーザ指定のハードウエアデバイスを接続する手段を具えている。

２）ＵＳＤとリアライザシステムロジック及び／又は相互接続チップとの間を接続している。ＵＳＤが、ロジックチップと類似する設計部の役割を果たすため、ロジックチップと同様の方法で、ＵＳＤＭを相互接続するのが好都合である。

２）通常、Ｌチップロケーションに取り付けられたロジックチップが行うように、ＵＳＤピンを相互接続ピンに自由に割当てる機能を設ける。

ユーザ指定のデバイスモジュールは、メモリモジュールがそのＲＡＭチップに有しているのと類似の機能を具えている必要があるので、ＵＳＤＭのアーキテクチャは、メモリモジュールのアーキテクチャと類似している。図３８は、ＵＳＤＭアーキテクチャを示している。ＵＳＤＭプリント回路基板、すなわち、ＵＳＤＭにプラグで取り付けられている移動可能なドーターカード（daughter card）の領域である、ユーザ指定のデバイス取り付け領域、又はマイクロプロセッサ、エミュレータ（emulator）計器と共通する方法で、ケーブルを介して接続されている他のこのような領域にデバイスを取り付ける。端末のブロックは、デバイス入出力ピンと、ＵＳＤＭロジックチップとの間に、コネクタ端末細条、一組のプリント回路ボードパッド、又は他のこのような手段を介して電気的な接続を行うための手段を具えている。端末のブロックは、デバイスの電源も具えている。物理的な、端末ブロックピンの容量が許す限り、１以上のデバイスを取り付けることができる。その代わりに、デバイスを、一般的な方法で、ケーブル及び中継装置を介して、遠隔的に接続することもできる。ＭＡ及びＭＤロジックチップの各々は、端末ブロックに接続されたＩ／Ｏピンと、相互接続に接続されたＩ／Ｏピンとを具えている。これらのチップを、メモリモジュールアドレス及びデータバスロジックチップにおいて説明したのと同様の方法で相互接続に接続する。付加的に、図にて示したように、メモリモジュールにチップを使用するのと同様の目的のために、これらのチップをホストインタフェースバス及び／又は共通コントロールバスにも接続する。一般的に、バスデータビットがＭＤチップに分配され、これによって、相互接続に分配されるように、ＵＳＤアドレス及びデータバスをＭＤチップに接続する。ＭＡチップをＵＳＤコントロールライン及び付加的にＵＳＤアドレスラインに使用する。図面は、可能性を説明するために接続された３個の仮説的ユーザデバイスを示している。ＵＳＤ０は、ＭＤチップを介して接続されたデータ及びアドレスバスと、ＭＡ０を介して接続されたコントロールラインＡ、Ｂ及びＣとを有している。ＵＳＤ１は、ＭＤチップに接続された３個のデータバスと、ＭＡチップを介してのアドレス及びコントロール接続とを有している。ＵＳＤ２は、アドレス指定のためのＭＡ１と、データのためのＭＤチップとを使用する。任意の特定のケースにおいて、リアライザシステムのユーザは、これらの設計及び使用にとって適切な方法を用いてこれらのＵＳＤを接続することができる。前記セクションにおいて示したように、メモリモジュールＭＤチップにおいて、双方向性ＲＡＭＤＱピンを相互接続するのと同様の方法を用いて、双方向性ＵＳＤ接続を相互接続する。相違点は、入力設計部の回路網を、出力イネーブルコントロールのようにして明示する必要があるという要件である。この回路網を、メモリモジュール数が２５及び２６の場合に示される“設計出力イネーブル”と同様の方法で相互接続ロジックに接続し、ＭＤチップの双方向性ドライバを制御する。通常、適切な出力イネーブルコントロール回路網を入力設計部中に設けていない場合、ユーザはこれを設ける必要がある。

【００５０】１.３.３.２　好適例のＵＳＤＭ

図３９において示した好適例において、ＲＡＭチップの代わりにＵＳＤを取り付けるための領域に関して、ＵＳＤＭはリアライザメモリモジュールと同一である。８個のＭＤチップの各々は１６個までのＵＳＤピンを相互接続し、２個のＭＡチップの各々は２３個までのＵＳＤピンを相互接続する。図は、２個の実際に取り付けられたＶＬＳＩデバイス、すなわちモトローラＭＣ６８０２０の３２ビットマイクロプロセッサ（"MC68020 32 Bit Microprocessor User's Manual"，Motorola，Inc.，Phoenix，1984）と、モトローラＭＣ６８８８１浮動小数点コープロセッサ（"MC68881 Floating Point Coprocessor User's Manual"，Motorola，Inc，Phoenix，1985）とを示している。これらのデバイスは、ＵＳＤの優れた例である。その理由は、一般的にこれらのデバイスをデジタルシステムの設計に用い、これらのロジック回路網表現をユーザが利用可能とすることはできないからである。これらのデバイスは、以下の入／出力ピンを有しており、この詳細については以下に示すとおりである。

ＭＣ６８０２０

データ　　：Ｄ３１—Ｄ０、双方向性

出力イネーブル条件：Ｒ／Ｗが"書込み"を示し、且つＤＢＥＮが真である場合、Ｄ３１—Ｄ０は、出力信号を送信し、そうでない場合には、入力信号を受信する。

アドレス　　：Ａ３１—Ａ０、出力

中央入力端子：ＣＬＫ、ＤＳＡＣＫ０、ＤＳＡＣＫ１、ＡＶＥＣ、ＣＤＩＳ、ＩＰＬ０—ＩＰＬ２、ＢＲ、ＢＧＡＣＫ、ＲＥＳＥＴ、ＨＡＬＴ、ＢＥＲＲ

中央出力端子：Ｒ／Ｗ、ＩＰＥＮＤ、ＢＧ、ＤＳ、ＤＢＥＮ、ＡＳ、ＲＭＣ、ＯＣＳ、ＥＣＳ、ＳＩＺ０、ＳＩＺ１、ＦＣ０—ＦＣ２

ＭＣ６８８８Ａ

データ　　：Ｄ３１—Ｄ０、双方向性

出力イネーブル条件：Ｒ／Ｗが"読出し"を示し、且つＤＳＡＣＫ０及び／又はＤＳＡＣＫ１が真の場合、Ｄ３１—Ｄ０は、出力信号を送信し、そうでない場合には、入力信号を受信する。

アドレス　　：Ａ４—Ａ０、入力

中央入力端子　　：ＣＬＫ、ＳＩＺＥ、ＲＥＳＥＴ、ＡＳ、Ｒ／Ｗ、ＤＳ、ＣＳ

中央出力端子　　：ＤＳＡＣＫ０、ＤＳＡＣＫ１

データバスとアドレスバスとをＭＤチップを用いて相互接続する。メモリデータバスのセクションにおいて説明したように、バスデータビットをクロスバーを横切ってスライスし、図に示すように相互接続を容易にしている。制御信号はＭＡチップによって相互接続される。出力イネーブル制御信号は、上述したように、制御信号に接続された特別のロジックによって発生される。ユーザは、このロジックを入力設計部に設け、設計部の残りの部分を用いてＬチップ中に実現する。各ＭＤチップが異なるＬ—Ｘバスの組を接続し、通常出力イネーブルコントロールが全バスに関して一般的なものであることから、設計部変換システムは、これらの回路網をＭＡチップの内のいずれか１個に接続するとともに、ＵＳＤＭコントロールバスを用いて、回路網を接続の必要性があるＭＤ及びＭＡチップに接続するように、ＭＡ及びＭＤチップを構成する。

【００５１】１.３.３.３　ユーザ指定のデバイスのための設計部の変換

ＵＳＤを特別の基本要素を用いて入力設計部内に設ける。ＵＳＤは、ユーザが作成するＵＳＤ仕様ファイルを示している特性データを伝達する。このファイルは、どのＬチップロケーションにこのデバイスを有するＵＳＤＭを取り付けるかを示すとともに、ＵＳＤのＩ／Ｏピンをリストする。この際、入力設計部のＵＳＤ基本要素中に使用されているピンネームを使用している。各ピンに対して、ＵＳＤは、ピンを接続しているＵＳＤＭロジックチップ及びピン数と、ピンが入力、出力又は双方向性であることとをリストする。ピンが双方向性である場合には、入力設計部中の出力イネーブルコントロール回路網のネームもリストする。設計部変換ソフトウエアシステムは、ＵＳＤを構成するとともに、ＵＳＤを設計部の残りの部分に接続するネットリストファイルを出力する。一般的な方法を使用するが、これには、以下のようなＵＳＤに対する例外がある：

・設計部リーダは、ＵＳＤ基本要素を設計部データ構造中に読出す。設計部リーダは、ＵＳＤ仕様ファイル中で読出しを行うために、ファイル特性を使用するとともに、後の使用のための基本要素記憶と関連する情報を記憶する。基本要素記憶を、各々異なる出力イネーブルコントロール回路網に接続された特別のピンに供給する。

・変換ステージは、構成が利用可能であり、且つ、ピンが正しく構成と対応していることをチェックする。

・システムパーティショナは、ＵＳＤをＵＳＤ仕様ファイルで特定されるＬチップロケーションに配置する。

・インタコネクタは、ＵＳＤピンに接続された回路網を特定のＬ—Ｘ相互接続バスに割当てる。インタコネクタはこれを行う際、ＵＳＤピンに接続された回路網を、ＵＳＤ仕様ファイルで特定されるＭＡ又はＭＤチップに接続するバスにのみ割当てることができ、且つ、イネーブルコントロール回路網ピンを、ＭＡチップに接続しているバスにのみ割当てることができるという制約を条件としている。

・ネットリストファイルをＵＳＤＭに送信するために：このＵＳＤＭのＵＳＤを制御している各々の出力イネーブルコントロール回路網が：基本要素を、この回路網のＭＡチップのネットリストファイルに送信する：その理由は、この回路網のために使用するＬ—Ｘバスを受信している入力バッファが、出力バッファの入力を駆動し、これによって、この回路網に割当てられたコントロール

バスラインを駆動するからである。このＵＳＤＭのＵＳＤに接続されている各回路網が：ＵＳＤ入力ピンを駆動する場合、基本要素を、このピンのロジックチップのネットリストファイルに送出する：その理由は、この回路網に使用される受信バスからの入力バッファが、このＵＳＤピンのために用いられる端末ブロックピンを駆動する出力バッファの入力端子を駆動するからである。ＵＳＤ出力ピンを受信する場合、基本要素を、このピンのロジックチップのネットリストファイルに送信する：その理由は、この回路網に用いられる駆動バスにつながっている出力バッファが、このＵＳＤピンに用いられる端末ブロックピンを受信している入力バッファの出力を受信する。ＵＳＤ双方向性ピンに接続している場合、基本要素を、このピンのロジックチップのネットリストファイルに送信する：その理由は、この回路網に用いられている受信バスからの入力バッファが、このＵＳＤピンに用いられる端末ブロックピンを駆動しているトライステート出力バッファのデータ入力端子を駆動し、この回路網に用いられている駆動バスにつながっている出力バッファが、このＵＳＤピンに用いられている端末ブロックピンを受信する入力バッファが一方の入力端子を駆動する２入力ＡＮＤゲートの出力を受信し、このピンの出力イネーブルコントロール回路網に割当てられたコントロールバスラインからつながっている入力バッファが、トライステート出力バッファのイネーブル入力端子及びＡＮＤゲートの他の入力端子を駆動するからである。

【００５２】１．４　構成

ロジック及び相互接続チップ技術のセクションにおいて説明したように、各チップの構成ビットパターンが、ＥＲＣＧＡネットリスト変換ツールによって出力される。リアライザ設計変換システムの最終ステージが、すべてのチップの発生する構成ファイルから設計部の単一バイナリー構成ファイルへと送られるデータを捕捉する。これによって、データがホストコンピュータ中に永久に記録される。リアライザシステムを各々使用する前に、構成ファイルからデータを読出し、このデータをホストインタフェースを介してリアライザハードウエアに伝達し、更にチップにロードすることによって、使用する設計部のロジックチップ及び相互接続チップを構成する。ホストインタフェースと、システム中のすべてのロジックチップ及び相互接続チップとの間に、構成接続を設ける。チップを構成すると、すべてのロジック機能及び相互接続の合計は入力設計部によって特定されるものと一致し、設計部が動作できる。好適例においては、Xilinx製のＬＣＡをロジックチップ及びクロスバーチップとして用いる。バイナリー構成ビットパターンを、１回に１ビットずつ、ＬＣＡ構成メモリのシリアルシフトレジスタにロードすることによってＬＣＡを構成する。各ビットを構成データ入力端子（ＤＩＮ）に供給するとともに、１回構成クロック（ＣＣＬＫ）を周期化してロードする。各ＬＣＡとホストインタフェースとの間の特別な構成接続は設けていない。その理由は、システムが全部で３５２０個までのロジックチップ及びクロスバーチップを具えなければならないからである。その代わりに、マルチビットデータバスと構成クロックとを有する構成バスを設け、これを、ＬＣＡを有するすべてのボードに接続する。構成を行うために、ループ毎に、データバス中のビット数と同数のチップを用いてロジックチップ及びクロスバーチップをグループ化する。１グループ中のすべてのチップを並列に構成する。図４０に示しているように、グループ中の各々のＬＣＡは、バスデータバスの異なるビットに接続された構成データ入力端子を有している。各グループにおける構成コントロールロジックブロックを、ホストインタフェースバスと、バス構成クロックと、グループ中のすべてのＬＣＡのクロック入力端子とに接続する。これらのコントロールロジックブロックを、ホストインタフェースバスを介して、ホストの命令によって選択的にイネーブルし、ホストインタフェースバスが意図するＬＣＡのグループがクロック信号を受信できるようにし、このような構成としている。これは、ホストコンピュータがリアライザシステムを構成するのに行う手続である。制御及びデータ伝送はすべてホストインタフェースを介して行われる：すべてのロジックチップ及びクロスバーチップを構成するために：各構成グループは：このグループのコントロールロジックブロックが、構成クロックをチップに送るように指示する。１個のＬＣＡ中の構成ビットと同数の周期の間：このグループ中の各チップの１構成ビットをバスデータバスにロードする。バス構成クロックを１回周期化する。次の周期へ。このグループのコントロールロジックが、もはや構成クロックを送信しないように指定する。次のグループへ。

【００５３】１．５　ホストインタフェース

ホストコンピュータの制御の下、リアライザシステムは周辺装置として動作する。ホストコンピュータは、設計部の構成ファイル中に記憶された構成ビットパターンを用いて、設計部に従ってリアライザシステムのロジックチップ及び相互接続チップを構成する。ホストコンピュータは、その外部リセット及びクロック信号を制御することによって連続的な設計部の動作を制御する。従って、ホストコンピュータは、スティミュレータ、サンプラ及びベクトルメモリを制御するとともに、ベクトル及び設計部メモリの内容を読出し及び書込むことにより、設計部と相互的に作用する。ホストコンピュータは、これらすべてをリアライザシステムホストインタフェースを介して行う。ホストコンピュータは、リアライザシステムのホストインタフェース及び構成バスを制御する。

【００５４】１．５．１　ホストインタフェースアーキテクチャ

リアライザシステムホストインタフェースを、完全に慣

用となっているラインに沿って構成する（図41）。リアライザシステムは、ホストインタフェースバスコントローラ、構成バスコントローラ、クロック発生器及びリセットコントローラを具えている。これらの各々について以下に説明する。インタフェースを、リアライザハードウエアシャーシのボードに構成するとともに、ケーブル及びインタフェースカードを介して、ホストコンピュータのI／Oバスに接続する。ホストインタフェースのコントロール機能を、特定のコンピュータの要求に応じて、ホストコンピュータのメモリアドレススペース、又は、入力—出力バススペースのいずれかに作成する。

【0055】1.5.2 ホストインタフェースバス

ホストインタフェースバスを、リアライザシステム中の、正規のロジックチップ及びメモリモジュールロジックチップの幾つか又はすべてのI／Oピンに接続する。ホストインタフェースバスは、リアライザシステムコントロール及びデータアクセス機能を割当てるアドレススペースを具えている。ホストは、最適なバスマスタであり、ホストインタフェースバスコントローラを介して、アドレス指定された読出しコマンド及び書込みコマンドをバスに送出する。ホストは、データをリアライザシステム機能とホストとの間に伝送する。ホストインタフェースコントロールロジックブロックを、メインロジックチップ及びメモリモジュールロジックチップにプログラムし、リアライザシステム機能を、バスを介して制御できるようにする。このバスによって制御される機能の特別な例としては、サンプラ、スティミュレータ、ベクトルメモリアドレス指定、オペレーション、ホストデータアクセス、及び設計部メモリホストデータアクセスがある。これらのコントロールブロックは、ロジックチップにすべてプログラムされているため、バスアドレススペース中の特別な機能及びロケーションを、すべてロジックチップのプログラミングによって規定し、任意の所定の設計部又は動作モードの必要に応じて変化させることができる。ホストインタフェースバスの特定の設計部は、特定のリアライザシステムを具体化した場合のデータアクセススピード及びハードウエアピンの利用可能性に依存している。好適例では、Rバスと称する11ピンホストインタフェースバスを、すべてのロジックチップの専用I／Oピンに接続している。好適例のハードウエアは、データ及びアドレスのために用いられる8個の双方向性ラインと、クロックと、2個のコントロールラインとを具えている。Rバスは32ビットアドレススペースと8ビットデータ幅とを有しており、ホストが40億までのユニークロケーションから8ビットデータを読出し又は書込みできるようにしている。Rバスを、アドレスレジスタ、データレジスタ及びコントロールレジスタを介して、ホストコンピュータにインタフェースする。これらを、慣用の方法でホストインタフェースバスコントローラによって構成し、ホストコンピュータのメモリ又は入力／出力スペース中に設ける。Rバスに接続される機能の例示：

1）一つのロケーションをRバスを介して書き込む際に、サンプリングクロックを周期化させ、ホストコンピュータのコマンドに従って、他のRバスロケーションからサンプリングされたデータの値を読出す、1グループとなっている8個のサンプラ。

2）ホストが特定のRバスロケーションに書込みを行う際に、データ値を変化させる、1グループとなっている8個のランダムアクセスメモリ。

3）各メモリロケーションを唯一のRバスロケーションに作成している設計部メモリ。ホストデータアクセスを行い、Rバスのアドレススペースへの読出し又は書込みオペレーションによって、ホストがアドレス指定された設計部メモリロケーションを読出し又は書込みすることができる。

他のこのような機能を容易に案出することができる。図42に、Rバスの動作を示す。ロケーションを読出すために、リアライザシステムを作動させるホストコンピュータでランするプログラムは、アドレスを、ホストインタフェースバスアドレスレジスタにロードするとともに、“読出し”コマンドビットをホストインタフェースバスコントロールレジスタにセットする。その後、ホストインタフェースバスコントローラは、Rバス読出しサイクルを作動させる。1回に8ビットずつ、各々Rバスクロックの周期で、アドレスはRバスデータラインに与えられる。第一サイクルの間、バスコントローラは“同期”Rバスコントローララインに、Rバスサイクルが開始していることを表明する。その後、“読出し”Rバスコントロールライン及びRバスクロックが5回目の周期を成し、アドレス指定されたバスインタフェースコントロールロジックブロックが、その読出しオペレーションを完成させることができる。Rバスクロックが6回目の周期を成す間、アドレス指定されたバスインタフェースコントロールロジックブロックが、読出しデータを8個のRバスデータラインに伝送する。バスコントローラはこのデータを捕捉し、ホストインタフェースバスデータレジスタにロードするとともに、“コンプリート”コマンドビットを、ホストインタフェースバスコントロールレジスタにセットする。“コンプリート”ビットを認識するホストプログラムをセットし、データを読出し、“コンプリート”ビットをクリアする。ホストプログラムが“書込み”コマンドビットをセットし、書込まれるべきデータをホストインタフェースデータレジスタにロードすることを除き、ロケーションの書込みも同様である。バスコントローラは、5回目のクロック周期において“読出し”Rバスコントロールラインを主張することはなく、6回目の周期においてデータをRバスデータラインに伝送する。この時、データがアドレス指定されたバスインタフェースコントロールロジックブロックによ

って捕捉される。ロジックチップ中に構成されるバスインタフェースコントロールロジックブロックは、上述した動作に従って、有限状態マシンと、完全に慣用となっている方法で制御された機能を用いてRバスを接続するデータバスとを具えている。

【００５６】１.５.３　構成バス

構成バス及びその使用とオペレーションとを構成セクションで説明する。バスは、ホストインタフェースを介してホストコンピュータによって制御される。バスを、データレジスタ及びコントロールレジスタを介してホストコンピュータにインタフェースさせる。これらのレジスタを慣用の方法でホストインタフェースハードウエアによって構成し、ホストコンピュータのメモリ又は入力／出力スペースに設ける。ホストコンピュータにおいてランする構成プログラムにって、構成バスデータレジスタにロードされるデータを構成バスデータバスに伝送する。ホストコンピュータが構成バスコントロールレジスタに書込みを行う場合、ホストインタフェースハードウエアは構成バスクロックを１周期させる。

【００５７】１.５.４　コントローラ及びクロック発生器のリセット

リアライザシステムのリセットコントローラは、２つのリセット信号を発生させる。システムリセット信号を、すべてのロジック及び相互接続チップのリセット入力ピンに接続する。ホストが主張する場合には、すべてのチップをリセットモードにし、構成の準備状態にする。慣用的な設計による１以上のプログラム可能なクロック信号発生器は、すべてのＬチップのＩ／Ｏピンに分配される出力信号を有している。ホストは、その出力周波数を制御し、サイクルを停止させること、再びサイクルさせること、特定回数サイクルさせること、連続的にサイクルさせること等が可能である。ホストは、リアライザシステムにおいて実現される設計部のクロック発生器として使用される。クロック信号を制御することによって、設計部のオペレーションを制御する。設計部のリセット信号を、すべてのＬチップのＩ／Ｏピンに接続する。設計部のリセット信号を、リアライザシステムにおいて実現される設計部をリセットする手段として用いる。これらの信号は、リアライザシステムによって実現される設計部との接続に利用することができる。特別な特性を入力設計ファイル中の回路網に組み込むことによって、入力設計部中の回路を、システムリセット又はクロックとして選定する。設計部リーダはこの特性を認識し、回路網を設計データ構造のリセット又はクロック回路網として特徴づける。設計変換システムの相互接続及びネットリスティング部分によって、この回路網をハードウエアの設計リセット信号又はクロック信号に接続されたＩ／Ｏピンに割当てる。

【００５８】２　リアライザ設計変換システム

リアライザ設計変換システムは、設計部リーダ、基本要素コンバータ、パーティショナ、ネットリスティング及び相互接続システム、ＥＲＣＧＡネットリスト変換ツール及び構成ファイルコレクタを具えている（図４３）。ここでは、入力設計ファイルを入力として用い、出力として構成ファイル及び対応テーブルファイルを作成する。これらは、リアライザハードウエアを構成、使用するための種々の応用に用いられる。入力設計を変換するために：

１）設計部リーダを用いて、設計部をメモリデータ構造に読込む。

２）設計部データ構造中の基本要素を、ホストＥＤＡシステム固有の基本要素から、ＥＲＣＧＡネットリスト変換ツールと適合するネットリストファイル中に送信することのできるロジックチップ基本要素へ変換する。

３）パーティショナを用いて、各構成要素をどのロジックチップに対して使用するかを決定する。

４）ネットリスティング及び相互接続システムを用いて、リアライザハードウェアシステム中の各ロジックチップ及び相互接続チップに対するネットリストファイルを出力する。

５）ＥＲＣＧＡネットリスト変換ツールを繰り返して使用することによって、各ネットリストファイルを、対応する構成ファイルへ変換する。

６）簡単な方法である構成ファイルコレクタを用いて、各ロジック及び相互接続チップの構成ファイルから、この設計部の単一構成ファイルへ送信される構成データを捕捉し、これを用いてリアライザハードウエアを構成する。

ここで説明した設計変換のための方法を、注意したことを除いて、入力設計部のロジックゲートとフリップフロップとの組合せの変換に用いる。これらの方法の変形例を用いて、特定目的の基本要素を変換する。これらの変形例については該当するセクションにおいて説明する。

【００５９】２.１　設計部リーダ

設計部リーダは、入力設計部ファイルを読出すとともに対応する設計部データ構造を構成する。

【００６０】２.１.１　入力設計ファイルの要件

ホストＥＤＡシステムによって作成される入力設計ファイルは、基本要素及びこれらの入出力ピンに関する記述と、２以上のピンを互いに相互接続するとともに、設計部の入出力端子と相互接続する回路網に関する記述とを有している。入力設計ファイルは、ネームのような、基本要素、ピン及び回路網に関連する情報も具えている。入力設計ファイルは、リアライザ設計変換システムが読出すことのできるように、基本要素の形態となっていなければならない。“基本要素”とは、ゲート、フリップフロップ又はメモリデバイスのような基本的ロジック素子である。設計者が特定することのできる基本要素によって規定される、より高レベルの構成を、リアライザシステムの読出し前に、ＥＤＡシステムによって構成基本

要素に変える必要がある。入力設計部において許容される一組の基本要素の一例としては、以下のMentor Graplrics Quick Sim基本要素がある。これは、好適例において読出される：

・２５個までの入力端子を有する簡単なゲート（ＢＵＦ，ＩＮＶ，ＡＮＤ，ＯＲ，ＮＡＮＤ，ＮＯＲ，ＸＯＲ，ＸＮＯＲ）

・特別ゲート（ＤＥＬ：ディレイ要素；ＲＥＳ：抵抗器；ＮＵＬＬ：開放回路）

・トライステート出力である無方向性伝送ゲート（ＸＦＥＲ）

・記憶デバイス（ＬＡＴＣＨ、レベル感応フリップフロップ又はＲＥＧ、クロックされたフリップフロップ）

・メモリデバイス（ＲＡＭ又はＲＯＭ）

【００６１】２．１．２　設計部データ構造

設計部リーダは、設計部データ構造を構成し、これを用いて基本要素を、ロジックチップネットリスティングに適した形態に変換し、基本要素をロジックチップサイズの区分に分割し、いかにしてロジックチップを相互接続するかを決定する。また、最終的には、設計部データ構造を、各リアライザロジックチップのネットリストファイルへと読出す。データ構造は、設計部の各基本要素、各ピン及び各回路網のレコードから成っている。各レコードは、関連に応じて、他のレコードに対するエンティティ及びリンク（すなわちポインタ）についてのデータを具えている。

・“基本要素”とは、ゲート、フリップフロップ又は、メモリデバイスのような基本ロジック要素である。

・各基本要素は、基本要素レコードによって表現される。基本要素レコードは、そのタイプ及び対象識別子のような基本要素に関するデータを有するとともに、他の基本要素とのリンクを有している。

・基本要素レコードは二重にリンクされたリストである。

・ピンとは、基本要素の入力接続又は出力接続である。

・基本要素のピンは、基本要素レコードと隣接して配置されるとともに、ピンネーム、ピンが反転されているかどうか、ピンの出力ドライブ等のピンに関するデータを有する、一連のピンレコードによって表現される。

・各基本要素は、一つの出力ピンのみを有しており、これを、任意のピンレコードとすることができる。

・“回路網”とは、相互接続されたピンの集合である。

・各回路網は、回路網レコードによって表現される。この回路網レコードは、その対象識別子のような回路網についてのデータを有するとともに、他の回路網へのリンクを具えている。

・回路網レコードを、二重にリンクされたリスト中に設ける。

・回路網のピンを、単一リンクの循環リスト中に設ける。

・各ピンレコードも、ピンの回路網へのリンクを有している。

・各回路網レコードは、回路網中の一つのピンへのリンクを有している。

図４４ａは、回路網の簡単な例を示しており、図４４ｂは、設計部データ構造を用いて回路網をどのように表現するかを示している。

【００６２】２．１．３　設計部リーダの方法論

設計部リーダは、入力設計ファイルから実現されるべき設計部を読出すとともに、対応設計部データ構造を構成することを目的としている。ここでの説明は、Mentor Graphics 設計ファイルに適合している。他も同様である。設計ファイルは、設計部中の各基本要素に対して、インスタンス（instance）と称するエントリを有している。設計ファイル中のインスタンスに取付けられた基本要素の特定のアスペクトについての情報が特徴である。以下に示す各工程の括弧内のネームは、好適例において用いられる実際のルーチンのネームである。

１）基本要素のレコードと、設計ファイル中の各基本要素に対するメモリ内データ構造中のピンのレコードとを以下のように作成する：各設計ファイルの基本要素の各インスタンスは：基本要素のタイプが何であるかを読出す（ｇｅｔ＿ｄｆｉ＿ｍｏｄｅｌ＿ｔｙｐｅ）。ユーザが規定したこの基本要素の配置についての情報が存在する場合には、これを“１チップ”特性から得る。；設計ファイルインタフェースを用いて、より高度な非基本的インスタンスをサーチする。このインスタンスはこの基本要素を具え、同様に特性を調べる（ｇｅｔ＿ｄｆｉ＿１ｃｈｉｐ）。インスタンスの各ピンは：ピンのネーム等の、ピンに関する任意の特性を捕捉する（ｇｅｔ＿ｄｆｉｐｉｎ＿ｉｎｆｏ）。次のピンへ。メモリ内設計データ構造中のレコードを、この基本要素及びピンに割当てる（ａｌｌｏｃ＿ｐｒｉｍ＿ａｎｄ＿ｐｉｎｓ）とともに、基本要素レコードを満たす。各々のピンは：ピンレコードを満たす。（設計ファイル中の接続された回路網の対象識別子ナンバを記憶し、トラックの識別子のナンバを、最大に維持する。）次のピンへ。次の設計ファイルインスタンスへ。ポインタのテーブル（ｎｅｔ＿ｔａｂｌｅ）を、ピンレコード(ピンポインタ)に割当てる。各構成し得る回路網に対するピンレコードに対象識別子ナンバで索引を付ける。最初はＮＵＬＬとする。上記最大識別子ナンバに従ってテーブルを作る。

２）各回路網のピンレコードをリンクし、以下のような、循環的にリンクされたリストとする：メモリ内データ構造中の各基本要素レコードにおいて：各ピンレコードは：‘ｉｄ’を、このピンの接続された回路網の対象識別子ナンバとすると、ｎｅｔ＿ｔａｂｌｅ〔ｉｄ〕が非ＮＵＬＬピンポインタを有している場合、これをこのピンレコードの“ｎｅｘｔ　ｐｉｎ”リンクへコピーする。このピンに対するピンポインタを、ｎｅｔ＿ｔａｂ

ｌｅ〔ｉｄ〕に入れる。次のピンへ。次の基本要素へ。
3）各回路網に対する回路網レコードを以下のように作成する：ｎｅｔ＿ｔａｂｌｅ中の各ピンポインタは：回路網レコードを割当てる。リンクを用いて、回路網レコードを、ピンポインタの指示するピンに接続する。対象識別子ナンバを用いてアドレス指定を行うことにより、設計ファイルインタフェースから回路網に関する情報を得る（ｄｆｉ＿ｇｅｔ＿ｎｅｔ,ｇｅｔ＿ｄｆｉ＿ｎｅｔ＿ｉｎｆｏ）。この回路網におけるピンレコードの循環リスト中の各ピンについて：回路網レコードに指示する。次のピンへ。循環リストを閉じる：最終ピンを最初のピンにリンクさせる。次のピンポインタへ。ｎｅｔ＿ｔａｂｌｅ記憶機能を解除する。
4）内部メモリ設計データ構造が終了し、設計部変換プロセスの後段が必要とする実現されるべき設計部についてのすべてのデータを表示する。

【００６３】２．２　基本要素コンバータ

基本要素変換は、Mentor Graphics Quick Sim 基本要素のような、ホストが特定する基本要素からの設計部データ構造中の基本要素を、ＥＲＣＧＡネットリスト変換ツールと適合させ、ネットリストファイル中に送出されるロジックチップ指定の基本要素に変換することを目的としている。この変換のいくつかは、簡易且つ直接的なものであり、単に、基本要素のタイプ及びピンネームのみを置換えるのみである。その他の変換はかなり複雑である。以下に示す特定の引用例は好適例のためのものであり、Mentor Graphics 入力設計ファイル中に存在するMentor Graplics Quick Sim のホストが特定する基本要素と、XilinxＬＣＡロジックチップが特定する基本要素とを使用する。設計部のゲートが、ロジックチップの特定するゲート基本要素中で許容されるよりも多くの入力端子を有している場合、このゲートを等価な機能を有するゲートの回路網で置換える。このゲート回路網の各々は許容数の入力端子を有している。このような置換を行うために、ゲートの基本要素レコード及びピンレコードを取り除き、新しいゲートの基本要素レコード及びピンレコードと、回路網中の新しい回路の回路網レコードとを加え、置換えられたゲートに接続されているピン及び回路網のピンレコード及び回路網レコードにリンクする（図４５ａ）。設計部のフリップフロップが、ロジックチップの特定するフリップフロップ基本要素において利用することのできない機能を有している場合、フリップフロップを等価な機能を有するゲートの回路網で置換える。まず第１に、回路網を解析し、機能を常に一定の値ではない回路網に接続するかどうかを調べる。例えば、ホストが特定する基本要素ＲＥＧを、常に一定の値ではない活動回路網に接続されたダイレクトクリア入力及びダイレクトセット入力の両方を用いて使用する場合に、メモリ内設計部データ構造における基本要素を、必要に応じて機能する７４７ＴＴＬフリップフロップロジックパートに用いられるのと類似のゲートの回路網で置換える。しかし、ダイレクトセット入力を、グランド回路網のような常にロジック値ゼロの回路網に接続する場合、すなわち、例えばグランド回路網に接続された１つの入力端子を有するＡＮＤゲートの場合には、ダイレクトクリアのみが実際に必要とされ、その代わりにロジックチップＤフリップフロップを代用する。Ｓ＿ＲＡＭ基本要素は、アドレス入力端子、双方向性データポート、読出しイネーブル及び書込みイネーブルを有するランダムアクセスメモリである。ＲＡＭ基本要素を、１以上のリアライザ設計部メモリモジュール中に作成する。基本要素変換ソフトウエアは、Ｓ＿ＲＡＭを利用可能な設計部メモリ構成と直接的にマッチする１以上のＸ＿ＲＡＭ基本要素に変換する。Ｓ＿ＲＯＭ（出し専用メモリ）基本要素は、イネーブル入力端子が存在せず、且つ、ＲＯＭの内容を含むファイルを付加していることを除き、Ｓ＿ＲＡＭと同様のものである。Ｓ＿ＲＯＭ基本要素を設計部メモリ構成と直接マッチする１以上のＸ＿ＲＯＭ基本要素に変換する。Ｘ＿ＲＯＭは読出しイネーブル入力端子を有しているが、書込みイネーブルを有していない。内容ファイルのパスネーム及びもとのＳ＿ＲＯＭに対するそのロケーションを、各Ｘ＿ＲＯＭ基本要素を用いて記憶する。リアライザハードウエアをこの設計部を用いて構成する場合、構成システムがパスネームを用いて、Ｘ＿ＲＯＭ内容を取り出すとともに、これらをホストインタフェースを介して設計部メモリにロードする。分離入出力データポートを有するＳ＿ＲＡＭは同様に操作される。しかし、これはMentor Graphics Quick Sim 基本要素中には設けない。オリジナル設計部のピン及び回路網は、初期特性すなわち“ｉｎｉｔｓ”を送信し、ある場合に持続的に幾つかの初期値を送信していることを示している。既知の値（０又は１）である持続的な初期特性がリアライザシステムによって観測され、ピン又は回路網を適切な“グランド”（すなわち、ロジック値０）又は“ＶＣＣ”（すなわち、ロジック値１）の回路網に接続する。特定のMentor Graphics の場合では：
：Ｔ、Ｘ、Ｒ及びＺの初期特性を無視する。ＯＳＦ（＝０＝０Ｓ）又は１ＳＦ（＝１＝１Ｓ）のみを観測する。
・回路網、すなわち、回路網の任意の出力ピンのＯＳＦ又は１ＳＦによって、出力ピンをグランド又はＶＣＣ回路網の一部分とする。
・入力ピンのＯＳＦ又は１ＳＦとによって、このピンを絶縁し、グランド又はＶＣＣ回路に接続する。

【００６４】オリジナル設計部の出力ピンは種々の強さのドライブを伝達し、シミュレータによって形成されるべき出力構造のタイプを示す。リアライザシステムは、基本要素変換の際に、幾分これらの強さを観測する。出力端子を、ハイのときにドライブの強さが零であり、ローのときにドライブの強さが強であるように特徴づける場合、出力端子はオープンコレクタとして識別され、こ

れを他の同様の出力端子及びレジスタに、ロジック設計者が“ワイヤード・アンド”回路網（図４５ｂ）と称する形態で接続するのが正当的である。同様に、ローのときにドライブの強さが零であり、ハイのときにドライブの強さが強である出力端子はオープンエミッタであり、“ワイヤードオア”を形成するのに用いられる。最終的にイネーブルされなければ、ＸＦＥＲ基本要素の出力ピンはドライブを有さず、他のＸＦＥＲ出力端子及びレジスタと配線され、“トライステート”回路網を形成する（図４５ｃ）。これらの構造のすべては、基本要素変換システムによって認識され、トライステート回路網のセクションにて説明したように、等価な機能を有する積の和のロジック回路網に変換される。特定のMentor Graphics の場合：

・Ｘ―ステートドライブの強さを無視する。

・１以上のＸＦＥＲ出力端子を回路網に接続することができるが、他の出力端子を接続することはできない。例外としては、入力ピンをグランド又はＶＣＣ回路網に接続しているＲＥＳ（抵抗器）を接続することもできる。ＸＦＥＲがイネーブルされない場合、回路網値はロジック値ゼロであり、ＶＣＣに接続されたＲＥＳを接続しない場合には、ロジック値１となる。１より多くのＸＦＥＲをイネーブルする場合、結果は論理的ＯＲとなる。

・０Ｃ／０Ｅ出力端子（ＳＺ／ＺＳ）は、同様のドライバを用いても駆動することのできる回路網のみを駆動することができる。駆動されていない場合、０Ｃ回路網はハイとなり、ＲＥＳを接続しているかどうかを無視して０Ｅ回路網はローとなる。

・ＲＺ、ＺＲ、ＲＳＳ、ＳＲ又はＺＺ出力ドライブを有する基本要素を、エラーなしで除去する。

・以下の出力回路網の条件によって致命的な誤りが生じる：すなわち、１より多くのストロング（strong）、ストロング及び抵抗器、１より多くの抵抗器、ＸＦＥＲ及びストロング、ＸＦＥＲ及びＳＺ，ＸＦＥＲ及びＺＳ、抵抗器を有していないＳＺ又はＺＳ、ストロングを有するＳＺ又はＺＳ、ＳＺ＆ＺＳである。

【００６５】Mentor Graphics ホスト及びXilinxＬＣＡを有する好適例の基本要素を変換するための特別な手続は、以下に示すとおりである（サブルーチンのネームが各ヘッダの後に続いている）：

１）ホストが特定する基本要素のＬＣＡ基本要素への初期変換（ｃｏｎｖｅｒｔ＿ｓ＿ｔｏ＿ｘ）。ホストが特定する基本要素は、上述のMentor GraphicsQuick Simセットから成り、‘Ｓ＿’プレフィクスを用いてネームが付けられる。ＬＣＡが特定する基本要素は、Xilinx.xnf仕様から成り、’Ｘ＿’プレフィクスを用いてネームが付けられる。各基本要素は：Ｓ＿ＩＮＶの場合、Ｘ＿ＩＮＶと置換え、ピンのネームを置換える。Ｓ＿ＢＵＳの場合、Ｘ＿ＢＵＳと置換え、ピンのネームを置換える。Ｓ＿ＲＥＳの場合、Ｘ＿ＢＵＦ、ＲＲドライブと置換え、ピンのネームを置換える。Ｓ＿ＤＥＬの場合、ＩＮ＆ＯＵＴ回路網を結合する。Ｓ＿ＡＮＤ、Ｓ＿ＮＡＮＤ、Ｓ＿ＯＲ、Ｓ＿ＮＯＲ、Ｓ＿ＸＯＲ、Ｓ＿ＸＮＯＲの場合、Ｘ＿ＡＮＤ、Ｘ＿ＮＡＮＤ、Ｘ＿ＯＲ、Ｘ＿ＮＯＲ、ＸＸＯＲ、Ｘ＿ＸＯＲと置換え、ピンネームを置換える。（２５ピンよりも多い場合、エラー）Ｓ＿ＲＥＧの場合、Ｘ＿ＤＥＦと置換え、ピンネームを置換える。Ｓ＿ＬＡＴＣＨの場合、Ｘ＿ＤＬＡＴで置換え、ピンネームを置換える。Ｓ＿ＸＦＥＲの場合、後にまでピンネームを後にまで残しておく。Ｓ＿ＮＵＬＬの場合、ピンネームをデリートする。Ｓ＿ＲＡＭ又はＳ＿ＲＯＭの場合、ピンネームを後にまで残しておく。次の基本要素へ。

２）初期特性の処理（ｇｅｔ＿ｉｎｉｔ）。メモリ内設計部データ構造中の２つの回路網は特別なものである。：すなわち“ｇｎｄ”（ロジック値０）及び“ＶＣＣ”（ロジック値１）である。各回路網は：回路網の初期特性がＯＳＦである場合、ｇｎｄ回路網が、初期特性がＯＳＦであることを認識できない場合、次の回路網へ、認識できる場合には、この回路網をｇｎｄ回路網と組合せ、次の回路網へ。回路網の初期特性が１ＳＦである場合、ＶＣＣ回路網が、初期特性が１ＳＦであることを認識できない場合、次の回路網へ、認識できる場合には、この回路網をＶＣＣ回路網と組合せ、次の回路網へ。各出力ピンは：ピンの初期特性がＯＳＦの場合には、ｇｎｄ回路網が、初期特性がＯＳＦであることを認識できない場合には次の回路網へ、認識できる場合には、この回路網をｇｎｄ回路網と組合せ、次の回路網へ。ピンの初期特性が１ＳＦの場合には、ＶＣＣ回路網が、初期特性が１ＳＦであることを認識できない場合には、次の回路網へ、認識できる場合には、この回路網をＶＣＣ回路網と組合せ、次の回路網へ。次のピンへ。次回路網は：ピンレコードを、リスト中に入れる。各入力ピンは：ピンの初期特性がＯＳＦであり、且つこの回路網がｇｎｄ回路網でない場合、ピンを回路網から切り離し、ｇｎｄ回路網に接続する。ピンの初期特性が１ＳＦであり、且つこの回路網がＶＣＣ回路網でない場合、ピンを回路網から切り離し、ＶＣＣ回路網に接続する。次のピンへ。次の回路網へ。

３）すべての出力ピンをチェックし、リアライザシステムのドライブの強さに影響を与えないで基本要素を取り除くとともに、常にイネーブル又はディゼーブルされているＸＦＥＲを取り除く。各基本要素は：出力ピンがドライブＳＳ、ＲＲ、ＳＺ又はＺＳを有していない場合、次のピンへ。出力ピンがＲＺ、ＺＲ、ＲＳ、ＳＲ又はＺＺを有している場合には、出力ピンを切り離し、除去する。出力ピンがＳ＿ＸＦＥＲである場合：Ｅ０（イネーブル）ピンが常にローである場合には、基本要素をデリートする。Ｅ０ピンが常にハイである場合、ＢＵＦを代用する。次の基本要素へ。

４）不法なマルチ出力接続を選別し、ワイヤードＯＲ，ワイヤードＡＮＤ及びトライステート回路網及びこれらのドライバを識別し、変換する(wired- nets)。各回路網は：ピンレコードを、リスト中に入れる。ＸＦＥＲ出力ピン、入力ピン及びｎｏｎ－ＸＦＥＲ出力ピンを数える。これらのピンは、ストロング（strong）かつレジスティブ（resistive）なＳＺ（オープンコレクタ）又はＺＳ（オープンエミッタ）である。ストロングを有する又はドライブ強度を有しない唯一の出力ピンの場合、次の回路網へ。１以上のレジスタを接続する場合、すべてのレジスタを、‘ＶＣＣ’（プルアップ）又は‘ｇｒｏｕｎｄ’（プルダウン）のいづれか一方に接続していることを確認し、いづれかを記憶する。以下の場合には、エラーであり、イグジット（exit）する：１より大きなストロング、１より大きなレジスタ。ＸＦＥＲ及びストロング、ＸＦＥＲ及びＳＺ、ＸＦＥＲ及びＺＳ。レジスタを有していないＳＺ又はＺＳ、ストロングを有するＳＺ又はＺＳ、ＳＺ及びＺＳ。１ストロング且つ１レジスタの場合には、レジスティブドライブを有する基本要素をデリートする。１より大きなＳＺの場合：（オープンコレクタ　ワイヤードＡＮＤ)各出力ピンは：レジスタの場合、出力ピンがプルアップであることを確認し、これをデリートする。レジスタでない場合には、このピンののドライブをストロングとし、Ｘ－ＩＮＶを構成し、この入力端子を出力ピンに接続するとともに、この出力端子を回路網に接続する。次のピンへ。回路網を“フローティングハイ”トライステート回路網として特徴付け、ＯＲ／ＮＯＲゲートを用いて、相互接続によってこれを構成する。１より多くのＺＳの場合：（オープンエミッタ　ワイヤードＯＲ)各出力ピンは：レジスタの場合：ピンがプルダウンであることを確認し、その後、これをデリートする。レジスタでない場合は、ピンのドライブをストロングにする。次のピンへ。回路網を“フローティングロー”トライステート回路網として特徴付け、インタコネクタによって、ＯＲゲートを用いこの回路網を構成する。０より多くのＸＦＥＲ且つレジスタ無し、又は、プルダウンの場合：（トライステート“フローティングロー”）各Ｓ－ＸＦＥＲは：ＡＮＤ　Ｉ０を構成するＸＦＥＲ　Ｅ０（又はＥＮＡ）及びＡＮＤ　Ｉ１を構成するＸＦＥＲ１０を用いて、Ｓ－ＸＦＥＲをＸ－ＡＮＤに変換する。次のＳ－ＸＦＥＲへ。任意のレジスタ基本要素をデリートする。回路網を“フローティングロー”トライステート回路網として特徴付け、ＯＲゲートを用いて、相互接続によってこれを構成する。０より多くのＸＦＥＲ基本要素且つ、プルアップの場合：（トライステート“フローティングハイ”）１個のＳ－ＸＦＥＲ基本要素の場合：ＮＡＮＤ１０を構成するＸＦＥＲ　Ｅ０（又はＥＮＡ）及びＮＡＮＤ１１を構成するＸＦＥＲ１０を用いて、Ｓ－ＸＦＥＲをＸ－ＮＡＮＤに変換し、反転する。１より多くのＳ－ＸＦＥＲ基本要素の場合：各Ｓ－ＸＦＥＲは：ＡＮＤ１０を構成するＸＦＥＲ　Ｅ０（又はＥＮＡ）及びＡＮＤ１１を構成するＸＦＥＲ１０を用いて、Ｓ－ＸＦＥＲをＸ－ＡＮＤに変換し、反転する。次のＳ－ＸＥＲへ、レジスタ基本要素をデリートする。回路網を“フローティングハイ”トライステート回路網として特徴付け、ＯＲ／ＮＯＲゲートを用いて、相互接続によってこれを構成する。次の回路網へ。

５）等価な機能を有するゲート回路網を用いて、ＬＣＡが特定するゲート基本要素中に許容されるよりも多くの入力ピンを有する任意のゲートを置換する。ゲート回路網の各々は、許容数の入力端子を有している。(wide-gates)各基本要素は：そのゲート及び入力端子が５よりも多く、２５以下である場合（XC3000ロジックチップを用いるものと仮定する）：同一種類の最終出力ゲートを構成する。その出力端子を、オリジナル出力端子及びコピー特性に接続する。必要とされる、より小さな入力ゲートの各々は：（ＡＮＤまたはＮＡＮＤオリジナル等のためのＡＮＤを用いて）ゲートを割当てる。ゲートの出力端子を最終ゲート入力端子に接続する。ゲートの入力端子を本来の入力端子に接続する。次のゲートへ。オリジナルのワイドゲートをデリートする。次の基本要素へ。

６）フリップフロップの機能をチェックするとともに、ＬＣＡの制約に合致すように配置する。XC3000シリーズを用いる場合、フリップフロップはダイレクトクリアを有するがダイレクトセットは有しておらず、従って両者を有するわけではない。すべてのＳ－ＤＥＦは、セット及びクリアのためのピンを有しているため、基本要素は、ピンを有しているといないとにかかわらず、置換える必要がある。XC3000はラッチをサポートしないため、ラッチを、等価なゲート回路網で置換える必要がある（ｆｌｏｐｓ＿ｆｏｒ＿３Ｋ)。各基本要素は：基本要素がＤＬＡＴ又はＤＥＦである場合：各ピンを記憶するとともに切離す。ＳＤ及びＲＤ回路網がゲートを介して、直接又は非直接的に‘ｇｒｏｕｎｄ’又は‘ＶＣＣ’であるかをチェックすることによって、ＳＤ及びＲＤが常にローであるかどうかを見出す。各基本要素がＤＬＡＴである場合：ゲートの回路網中に組込み、必要な場合にのみ、ＳＤ及び／又はＲＤのためのゲートを具えるラッチを構成する。オリジナル基本要素及びピンレコードをデリートする。各基本要素がＤＬＡＴでなく、ＤＥＦである場合：ＳＤが常にローの場合、ＳＤを用いずにＸ－ＤＥＦを構成し、これを接続する。ＳＤがローでなく、ＲＤがローの場合、入力端子及び出力端子にＸ－ＩＮＶを用いてＸ－ＤＥＦを構成し、これを接続し、Ｘ－ＤＥＦのＲＤピンを、ＳＤ回路網に接続する。ＳＤが常にローではない場合、６個の３－入力ＮＡＮＤ及び２ＩＮＶの回路網に組込み、TTL7474 と同様に、セット及びクリアを有するＤＥＦを構成する。オリジナルの基本要素をデリートする。次の基本要素へ。

7）S－RAM及びS－ROMを、X－RAM及びX－ROMに変換する。各基本要素は：基本要素がS－RAM又はS－ROMである場合：そのハイト（height）（ワード数）を、アドレスピン（ハイト＝２～ピンカウントの累乗）及び、データピンナンバと等しいアドレピンの幅をカウントすることによって決定する。各利用可能な設計部メモリ構成は：S－RAM／ROMハイトを、設計部メモリハイトで割算し、必要なモジュールの行数を得る。S－RAM／ROMの幅を設計部メモリの幅で割算することにより、必要なモジュールの列数を得る。この構成のために必要なモジュールの総数は行掛ける列である。次の構成へ。必要なモジュール数が最小となる構成を選択する。行よりも多くのモジュールを必要とする場合、モジュールの各行に対する出力端子とハイオーダのアドレス回路網に接続された入力端子とを用いて、デコーダの基本要素及び回路網を構成する。各行は：（X－RAMのみ）２個の入力端子を用いて書込みイネーブルのためのANDゲートを構成する：２個の入力端子とは、この行のデコーダ出力端子及びS－RAM書込みイネーブルである。２つの入力端子を用いて、行読出しイネーブルのためのANDゲートを構成する：この２個の入力端子とは、この行のデコーダ出力端子及びS－RAMリードイネーブルである。次の行へ。モジュールの各行について、：各コラムは：X－RAM／ROM基本要素を構成するとともにその構成を記憶する。X－ROMの場合、そのファイルネーム及び行数及び列数を記憶する。X－RAM／ROMの読出し及び書込みイネーブルピンを、この行の（X－RAMのみの）読出し及び書込みイネーブルピンに接続（又は、この行が１つのみの場合には、S－RAMのイネーブルピンに接続）する。X－RAM／ROMのアドレスピンを、より低オーダのアドレス回路網に接続する。X－RAM／ROMのデータピンを、この列に対応する一組のデータピンに接続する。次の列へ。次の行へ。オリジナルのS－RAM／ROM基本要素をデリートする。次の基本要素へ。

【００６６】２．３　パーティショナ

リアライザハードウェアはユニット及びサブユニットの階層から成っている。すなわち、ボードは論　理チップを具え、ボックスはボードを具え、ラックはボックスを具える等である。各ユニットは、ユニット固有のロジック及び他のユニットとの相互接続のための容量を有している。実現すべき設計部をこの階層に応じて区分化し（すなわち、サブ分割し）、基本要素の多重クラスタとする。各ボックスのロジック及び接続容量に応じて作成された一組のボックス区分を設ける。これらの区分の各々を、ボードのサブ区分に分割する等により、単一の論理チップにプログラムするのに十分な程小さな区分に分割する。同様の分割化方法を、階層の各レベルに順次適用する。分割化の目的とは：

1）各基本要素を、ボックス、ボード及びロジックチップに割当てること、

2）ユニット（ボックス、ボード又はロジックチップ）の相互接続能力の下、区分と接続している回路数を保持すること、

3）ユニットの限界内で分割化に用いられるロジックの量を保持すること、および

4）区分の総数すなわち使用されるユニットの総数を最小にすることである。

【００６７】２．３．１　分割化方法

ここで説明する好適分割化方法は、“カット（CUT）回路網”（クラスタの外部における基本要素の接続）の数が最小であり、お互いに高密度で相互接続されたロジック基本要素を密集させるプロセスに基づくものである。各クラスタは、ボックス、ボード又はLチップに対応する区分である。前記プロセスは、以下において指摘する、相当な改良を伴うPalesko 及びAkers の従来の区分化方法(Chet A. Palesko, Lex A. Akers, “Logic Partitioning for Minimizing Gate Arrays”,IEEE Trans, CAD, NO.2, pp. 117～121, April 1983)に基づいている。すべての基本要素を初めから見えているクラスタに、割当てられていない基本要素から成る“ナル（null）クラスタ”を設ける。まず、ナルクラスタからシード（seed）クラスタを選択し、その後、これをリピートし、すべてのナルクラスタ基本要素の“利点”を計算し、最も大きな利点を有する基本要素を選択することによって各クラスタを形成する。基本要素の利点が大きくなるにつれて、ロジッククラスタに組込むのに適したものとなる。

【００６８】２．３．２　有利な機能

部分的な利点とは、この基本要素をクラスタに組込む場合にこのクラスタのカット回路網の数がどのように変化するかに基づくことである。ユニットを最大相互接続可能性以下に保持するために、クラスタのカット回路網総数をカウントする必要がある。基本要素のピンを具えている各回路網は垂直となっており、基本要素を組込むものを仮定すると、閉回路網、“多数カット”の回路網又は“単一カット”の回路網のいずれかに分類される。一つのみの接続をクラスタの内側に設けると単一カット回路網であり、１より多くの接続をクラスタの内側に設けると多数カット回路網となる。閉回路網は、全体がクラスタ中に含まれる回路網をいう。図４６は、クラスタと、３個の回路網S，M及びEによって接続された５個の基本要素とを示すとともに、影を付けた基本要素をクラスタ中に移動させるとどうなるかを示している。クラスタのカット回路網数を１個増加させると、回路網Sは、単一カット回路網となり、カット回路網数を１個減少させると、回路網Eは、閉回路網となる。回路網Mは、クラスタのカット回路網数を増加、減少させても多数カット回路網であり、このため無視される。クラスタ回路網の変化は、単一カット回路網と閉回路網との差で

ある。すなわち：
〔クラスタカットの変化〕＝〔単一カット回路網〕－〔閉回路網〕
好ましい有利な機能とは、各基本要素の個数を定め、クラスタに組込むには、どの基本要素を選択するのが最適かを決定することである。最大数のピンと最も強固に接続された基本要素を選択するのが最適である。この機能は、Palesko andAkers の分割化に関する最初の有利な機能である。すなわち：
〔クラスタカットの変化〕＞０の場合：
〔利点〕＝〔基本要素のピン数〕／〔クラスタカットの変化〕
〔クラスタカットの変化〕≦０の場合：
〔利点〕＝〔－（クラスタカットの変化）＊１００〕＋１００＋〔基本要素のピン数〕
この基本要素をクラスタに組込む場合、クラスタカットの数が増加する。多くのピンを具えれば具える程、加えるカット回路網の数が少なければ少ない程優れている。クラスタのカット数が減少する場合、減少の程度は１００倍となり、１００が加えられ、カットを減少させない、いかなる基本要素の利点よりも大きな利点が得られることを保証している。クラスタカットが減少している場合、ピンの数を増やし、接続を断ち切ることにより、基本要素を、更に多くのピント結び付ける。好適な方法で用いられる改良とは、クラスタカットが増加する場合に、ピン数の項をピン数／カット変化の比に加えることである。この変更は、前記比が等しい場合に、より多くのピンを有する基本要素を選択することによって、初期シード選択を改善することができる。前記比を１０倍することにより、ピン数単独よりも効果的にする。このことは、好適且つ有利な機能である。すなわち：
〔クラスタカットの変化〕＞０の場合：
〔利点〕＝（〔１０＊（基本要素のピン数）〕／〔クラスタカットの変化〕）＋〔基本要素のピン数〕
〔クラスタカットの変化〕≦０の場合：
〔利点〕＝〔－（クラスタカットの変化）＊１０００〕＋１００＋〔基本要素のピン数〕
【００６９】２.３.３　クラスタの構成
初めに、すべての基本要素をナルクラスタ中に配置する。ユーザは、選択するＬチップ、ボード等を表示する特性を入力設計部に付加することによって、基本要素を特定のクラスタ中に予め配置することができる。この時、これらの予め配置された基本要素は、クラスタ情報に関数するシード配置としての役割を果たす。このことによって、ユーザは、タイミング感知基本要素又は他の高優先基本要素を集めることができ、また、高優先基本要素に強固に接続されている他の基本要素を集めることによって、分割化の結果を変えることができる。各々の新しいクラスタに関して、始めに、新しいクラスタの未配置の基本要素の利点を計算し、基本要素レコード中に記録する。予め配置を設けない場合、最も有利な基本要素（すなわち、最も大きな利点を有するもの）を、クラスタの初期シード基本要素として選択する。最も有利な基本要素の各々をクラスタ中に移動させた後、組込まれた基本要素と同一の回路網におけるピンを有する基本要素のみが、利点を再び計算する。他の基本要素は移動によって影響を受けないので、クラスタのこれらの利点に変わりはない。従って、クラスタが一杯になるまで、新しい最も有利な基本要素をクラスタ中に移動させる。クラスタが一杯となる時を決定することは、ロジック容量及び相互接続（すなわち、クラスタカット回路網）の両方に依存している。基本要素を、クラスタ中に移動させる場合、常に、基本要素によってクラスタ中のゲート数が増加する。しかし、基本要素によって、カット回路網は、必ずしも増加しない。減少することもあり得る。Palesko and Akers の方法による相互接続の限界に達する場合、基本要素がロジック容量又は相互接続の限界を超えないならば、最大よりも少ない利点を有する基本要素をクラスタ中に移動させることができるが、局所的相互接続の最大を超える場合には基本要素をクラスタ中に移動させることはできない。ここで説明した方法を、以下の点において改良する。すなわち：マーカ（marker）のアレイを設ける。各マーカに対して、マーカは、起動可能である。基本要素を１個ずつクラスタ中に移動させる。各々の移動の後、クラスタカット回路網の数をチェックする。クラスタカット回路網の数がユニットの最大利用可能相互接続機能以下の場合、移動を、相互接続が可能なものとして認識される。最大ロジック容量の限界に達した場合、最後の移動は、相互接続可能なものとは認識されず、最後の移動が相互接続可能となるまでは移動は行われない。ユニット（ラック、ボックス又はボード）をサブユニット（ボックス、ボード又はＬチップ）に分割化するために：予め配置されていないすべての基本要素をナルクラスタ中に移動させる。各クラスタは：各ナルクラスタ基本要素の利点を計算するとともに記憶する。起動カウンタの数をゼロにする。〔クラスタ基本要素カウント〕＜〔最大ロジック容量〕の場合、移動カウンタをインクリメントする。最も有利な基本要素をクラスタ中に移動させる。どの基本要素を移動させるかを、移動カウンタに記録する。〔クラスタカット回路網〕＜〔最大相互接続容量〕の場合、ｍｏｖｅ〔移動カウンタ〕＝ＯＫをマークする。〔クラスタカット回路網〕≧〔最大相互接続分量〕の場合、ｍｏｖｅ〔移動カウンタ〕＝ＮＯＴ　ＯＫをマークする。このクラスタに接続された回路網の利点を計算する。次のリピートへ。ｍｏｖｅ〔移動カウンタ〕＝ＮＯＴ　ＯＫの場合、ｍｏｖｅ〔移動カウンタ〕に記録された基本要素をクラスタから外へ移動させる。移動カウンタをデクリーメントする。次のリピートへ。次のクラスタへ。区分化のプロセスは、すべての基本要素を首尾よくクラスタ中に配置す

るまで又はすべてのクラスタが一杯となるまで続き、プロセスが終了する。好適例の全設計部を区分化するために：ラッチレベルのボックス、すなわち、各ボックス毎に１個のクラスタに分割化する。この際：
〔最大ロジック容量〕＝〔全ボックス及び最大相互接続容量〕＝〔ボックス毎のＹ－Ｚパス〕
を用いている。各ボックスクラスタは：ボックスレベルのボードへの区分化、すなわち、各ボード毎に１個のクラスタに分割化する。この際：
〔最大ロジック容量〕＝〔全ボード及び最大相互接続容量〕＝〔ボード毎のＸ－Ｙパス〕
を用いている。次のボックスクラスタへ。各ボードクラスタは：ボードレベルのＬチップへの区分化、すなわち、各Ｌチップ毎に１個のクラスタに分割化する。この際：
〔最大ロジック容量〕＝〔Ｌチップ及び最大相互接続容量〕＝〔Ｌチップ毎のＬ－Ｘパス〕
を用いている。次のボードクラスタへ。

【００７０】２.３.４　容量の限界

この方法に用いられる最大ロジック容量の限界の決定は、使用するロジックチップの特性に依存している。Xilinx製のＬＣＡを論理チップとして使用する場合、これらは、構成可能な論理ブロック(ＣＬＢ）を基礎にしている。ＣＬＢの各々によって、多くのゲート及びフリップフロップを具現化することができる。ＣＬＢの数は、ゲート及びフリップフロップ機能、どの位多くの機能を設けるか、どのくらい多くのピンを有するか、どのように相互接続するかに依存している。分割化前に設計部をＣＬＢの形態に変換する場合、ＣＬＢを分割化された基本要素とし、ロジック容量の限界はＬＣＡ中のＣＬＢの数に基づいている。分割化前に設計部をＣＬＢの形態に変換しない場合、ゲートを、分割化された基本要素としてロジック容量の限界は、ＬＣＡに適合すべきゲートの数に基づいている。ゲートが消費する容量の程度に応じて、ゲートの重みをかけ、分割化の結果を改善する。各クラスタを構成するのに用いられる限界は、必ずしもすべて同一である必要はない。ユニット間でロジック及び相互接続容量特性を変える場合、適切に限界を設定してこれらユニットのクラスタを構成する。

【００７１】２.３.５　リアライザの区分化

プロセスを分割化することによって、設計部の各基本要素に対する３ナンバボックス／ボード／チップロケーションが得られる。このロケーションは、設計部データ構造の基本要素レコードに記憶される。このことによって、設計部回路網の各基本要素は、Ｌチップ、ボード、及びボックスに亘ってトレースすることができる。ネットのタイミングは、システムにおける回路網をトレースするとともに、相互接続クロスバーチップ及び論理チップを介して、ディレイを加算することによって評価される。相互接続の段階では、回路網中に具えられた種々のボックス／ボード／チップ基本要素の組合せ総数に基づき、ネットリストをオーダする。このようにして、相互接続は最も複雑な回路網から最も複雑でない回路網までを保証する。最終的に、回路網及び回路網レコードの基本要素が、Ｌチップ及びクロスバーチップに亘って回路網を明確に作成する情報を具えているので、局所的な図式的ロジック変化を再度区分化する必要はなく、変化させられた回路網を具えるチップを更新する必要があるだけである。これによって、設計部を再度区分化せずに設計部を変換させることができる。

【００７２】２.４　ネットリスティング及び相互接続システム

リアライザネットリスティング及び相互接続変換システムは、入力設計に応じて、リアライザハードウェアを構成するのに用いる、リアライザシステム中の各ロジックチップ及びクロスバーチップに関するネットリストファイルを構成することを目的としている。どのようにして部分的クロスバー相互接続をネットリストすべきかの決定を、次の３段プロセスの総合によって行う。

ステージ１：ステートメントは、各基本要素毎に設計データ構造中のすべてのロジック基本要素に関するロジックチップネットリストファイルに送信される。

ステージ２：完全に単一ロジックチップ中に含まれているトライステート回路網の加算ゲートに関するステートメントは、各回路網毎に送信される。

ステージ３：より多くのロジックチップ間を通る回路網の相互接続をネットリストする。各カット回路網毎に、すべてのチップにおけるこの回路網のすべての相互接続バッファ及び、クロスバーチップにおけるこの回路網の加算ゲートに関するステートメントを送出する。このプロセスの一部として、いかにして回路網を相互接続するかを明確に決定する。このプロセス自体は４個のステージを有している。

ステージ３ａ：どのようにして回路網が各クロスバー間を通り、且つ、どこにロジックチップドライバ及びレシーバを配置するかを示しているトリー（tree）を構成する。

ステージ３ｂ：クロスバーチップの各組の回路網を相互接続する能力を評価する。

ステージ３ｃ：この回路網を相互接続するクロスバーチップの最適な組を選択する。

ステージ３ｄ：組の選択及びトリー構造に基づき、バッファ及び加算ゲートに関するステートメントをロジック及びクロスバーチップネットリストファイルに送出することによって、相互接続をネットリストする。このセクションでは、各ステージに用いられる技術について説明しており、完全な相互接続及びネットリスティング手続について詳細な規定と、二つの詳細な回路構成例を記述している。

【００７３】２.４.１　シンプルな回路網及びトライス

テート回路網の相互接続構造

シンプルな回路網は、単一のドライバのみを有する回路網である。ドライバを有するソースLチップは、信号を、階層の上方へ向かってすべてのレシーバに及んでいるクロスバーチップに伝達する。レシーバを駆動するためのパスを階層の下方に向かって接続し、すべての受信Lチップを駆動する。図47は、シンプルな回路網の相互接続を示しており、詳細については以下に説明する。

トライステート回路網は、2以上のトライステート、オープンコレクタ又はオープンエミッタドライバによって駆動される回路網である。このことは、設計部データ構造中においては、2以上のドライバ（出力ピン）を有する単一の回路網として示される。各ドライバを、基本要素変換の間にドライバを変換して得られる1個のANDゲート及び1個以上のレシーバ（入力ピン）である。ドライバがイネーブルされない場合にゼロとなっている“フローティングロー”回路網を、1個以上の加算ORゲートをANDゲートにより駆動することで実現する。“フローティングハイ”ゲートは、ANDゲートに反転データ入力を有しており、最終的な加算ゲートをNORとしている。同じ接続形態及び基本的な方法を、両ケースに適用する。一般的な一方向性接続及び1以上の加算ORゲートを用いて、積の和としてトライステート回路網を具体化する。ドライバのパスを、相互接続階層のXからZに向かって集中させ、ドライバを加算ORゲートに集める。最も高いレベルの加算ORゲート出力を、ロジック回路網の真の値、すなわち、そのソースとしている。ソースを相互接続階層の下方に向かって接続し、すべてのドライバを駆動する。結果的に、幾つかのチップ対（Z—X、X—Y及び／又はY—Z）は、2つのパスを必要としている。そのうち、一方のパスはドライバを加算ORゲートとつないでいるものであり、レシーバと出力とをつなぐものである。図48は、トライステート回路網の相互接続を示しており、詳細については以下に説明する。

【0074】2.4.2 ネーミング

固有のネームを有する回路網を用いて、論理チップ内の相互接続をネットリストファイル中に規定する。これらの回路網は、設計部データ構造中の回路網と混同すべきではない。各設計部回路網は、論理ロジックチップネットリストファイル中に片方の回路網を有し、入力設計ファイルに用いたのと同じ実際の回路網ネームをネットリストファイルに用いる。基本要素変換の間に設計部データ構造に加えられる回路網に、人工的に発生させられたネームを付ける。設計部データ構造中に存在しない回路網を、ロジックチップ及びクロスバーチップネットリストファイルに送出し、相互接続を特定する。ロジックチップ又はクロスバーチップ I／OバッファとI／Oピンとの間の回路網、ANDゲートとトライステートの積の和である加算ゲートとの間の回路網、及び、クロスバ一加算を用いる場合に相互接続の上方及び下方に通っている回路網、これらすべての回路網は設計部の単一回路網と関連しているが、ネットリストファイル中の回路網とは別個のものである。相互接続基本要素をネットリストファイルに送出する際は、実際の回路網ネームを変更し、これら相互接続機能の各々に対して別個の回路網ネームを提供する。次のチャートはネーム変更のすべてをリストしている。I／Oバッファとそのピンとの間のチップレベル毎に、1個のネームのみを使用する。これらのネームには、接続の他端におけるチップに従って番号を付して独自性を持たせている。チップレベル毎に1回よりも多く使用されるネームによって、クロスバーチップ内部接続を規定する。これは、このような多くの構成し得るネーミングシステムの一例にすぎない。文字‘H’を、チャート中の実際の回路網ネームの代わりに用いる。例えば、相互接続された回路網を‘ENABLE’と称する場合、ロジックチップ6から受信する入力バッファ入力端子とそのI／Oピンとの間の回路網を‘ENABLE—D—6’と称する。

‘N’：

Lチップ：このLチップを回路網のソースとする場合、真の回路網値である。このLチップに唯一つのドライバを設ける場合トライステートドライバである。

X，Y，Zチップ：1個のチャイルド（child）ドライバを設ける場合、チャイルドからの入力バッファ出力ピンである。このチップを回路網のソースとする場合、チャイルドへの出力バッファ入力ピンである。

すべてのチップ：親への出力バッファ入力ピンである。加算ゲート出力端子。

‘N_R’：

Lチップ：この回路網のソースをいずれかに設ける場合、真の回路網値である。

X，Y，Zチップ：このチップが回路網のソースではない場合、チャイルドの出力バッファの入力ピンである。

すべてのチップ：親からの入力バッファ出力ピン。

‘N_R_c’：

X，Y，Zチップ：チャイルドへの出力バッファ出力ピンである。ここで‘c’は、チャイルドのチップナンバである。

‘N_P’：

すべてのチップ：親からの入力バッファの入力ピン。

‘N_D’：

すべてのチップ：親への出力バッファの出力ピン。

‘N_D_c’：

X，Y，Zチップ：チャイルドからの出力バッファの出力ピン。ここで‘c’はチャイルドのチップナンバである。

‘N—P’：すべてのチップ親からの入力バッファの入力ピン。

‘N—D’：すべてのチップ親への出力バッファの出力

ピン。
‘N－D－c’：X，Y，Zチップチャイルドからの入力バッファの入力ピン。
‘N_OR_i’：
Lチップ：1より多くのドライバをLチップに設けた場合、トライステートドライバである。ここで‘i’は多くのこのようなドライバを識別している。
X，Y，Zチップ：1より多くのチャイルドドライバを設ける場合、チャイルドドライバからの入力バッファ出力ピン。
すべてのチップ：加算ゲート入力端子。
【００７５】２.４.３　ステージ１：ロジック基本要素のネットリスティング
ステートメントは、各基本要素毎に、設計部データ構造中のすべてのロジック基本要素に対するロジックチップネットリストファイルに送出される。基本要素を接続している回路網のネーミングを行い、以下のステージ３ｄの相互接続バッファに用いられるネーミングと一致させる。回路網のソースを同一の論理チップ中に設ける場合、入力ピンをこれらの本来の回路網ネームに接続する。このことは、閉回路網（カットがない回路網）に対して常に真であり、カット回路網のLチップを駆動する際においても真である。Lチップをソースとしない場合、入力ピンをこれらの親レシーバの入力バッファに接続する。出力ピンをロジックチップの加算ゲートに接続する場合を除き、出力ピンをこれらの本来の回路網ネームに接続する。出力ピンをロジックチップの加算ゲートに接続する場合、特定の回路網ネームを変更させる。
【００７６】２.４.４　ステージ２：ロジックチップ加算ゲートのネットリスティング
完全に単一ロジックチップ中に含まれているトライステート回路網の加算ゲートに関するステートメントを、各回路網毎に送出する。上述した回路網ネームの変更を用いて入力端子を接続する。出力端子は本来の回路網ネームを駆動する。回路網が“フローティングハイ”であるかどうかに応じて、適切な出力検知（ＯＲ又はＮＯＲ）を用いる。
【００７７】２.４.５　ステージ３：カット回路網相互接続の決定及びネットリスティング
1より多くの論理チップ間を通っている回路網（カット回路網）の相互接続をネットリストする。カット回路網を、各々、ステージ３ａ、３ｂ及び３ｃを介して一度に処理する。
【００７８】２.４.５.１　ステージ３ａ：相互接続トリーの構成
一時的なトリーデータ構造を構成し、相互接続プロセスを案内する。この一時的トリーデータ構造は、この回路網に基本要素を有するLチップと、相互接続を具体化するX、Y及びZチップと、各々の相互接続の要件を示すことによって、回路網の構造を表現する。各トリーレベルでの各ノードはシステム中のロジック又はクロスバーチップに対応しており、その下位のチャイルドノードにつながっているブランチを有し、ノード及び親のパスにつながっている相互接続パスに関するデータを以下のように記憶する：

レベル	チップ	相互接続パス
ルート	Zチップ	なし
第１レベル	Yチップ	Y－Zパス
第２レベル	Xチップ	X－Yパス
第３レベル	Lチップ	L－Xパス

回路網中に含まれている各Lチップは、回路網にいかに多くの基本要素を有していても、トリー中の唯一個のノードで表現される。各ノードは以下のエントリを有している。
チップナンバ：ボードのいずれかのLチップ、ボックスのいずれかのボード又はラックのいずれかのボックス。初期値はＮＵＬＬ。
D及びRカウント：このノードのパスに必要とされるドライバ（D）及びレシーバ（R）の数である。初期値はゼロ。
D及びRパス：（各L－X、X－Y又はY－Zパスで利用できるいくつかのパスナンバの中から）いずれかのパスナンバを用いて、ドライバがこのノードからトリーを上り、レシーバが降りる。初期値はＮＵＬＬである。
トップサム：このノードが下位にすべてのドライバを具える加算ゲートを有している場合、真と認識する。これを用いて、多数ゲートの積の和における最終ゲートを制御し、“フローティングハイ”の場合その出力反転を得る。初期値は偽である。回路網が多数のボックスに及んでいない場合、ルートノード（root node）はナルエントリ及び唯一つの第１レベルノードを有している。回路網が多数ボードに及んでいない場合、第１レベルノードはナルエントリ及び唯一つの第２レベルノードを有している。回路網が多数のLチップに及んでいない場合、回路網は、相互接続の必要がなく、トリーを有していない。パーティショナによって割当てられた基本要素のロケーションに従って、設計部データ構造中の回路網を操作しトリーを構成する。回路網が１より多くのボックス又はボードに及んでいない場合、必要とされないクロスバーレベルのノードをナルとする。このようにして、各Lチップの駆動出力端子及び受信入力端子の数をカウントするとともにLチップノード中に記憶し、Lチップの相互接続の必要性を確認する。各Xチップノードにおいて、ドライバを有しているLチップの数及びレシーバを有している数をカウントし、各Xチップが、どのような相互接続を設けなければならいかを確認する。同様に、各Yチップにおいて、駆動及び受信Xチップをカウントし、Zチップにおいて、Yチップをカウントする。最後に、トリーを分析し、ソースである回路網の真の値をレシーバに伝送するポイントを決定する。簡単な回路網で

は１個のＬチップの中にソースを設けている。クロスバー加算を用いているため、ソースをトライステート回路網のクロスバーチップとすることもできる。通常、クロスバーチップがチャイルドチップ間にレシーバを有している場合、クロスバーチップをネットリストし、真の値を、そのより高レベルのペアレントチップから送信する。しかし、階層中において、親チップより下位のチップがソースを有している場合、親チップは、真の値を親チップ自体又はそれより下位のチップから得る。このようにするために、クロスバーノードを走査し、ノード又はノードの派生がソースの場合にはレシーバカウントをゼロにセットする。

【００７９】２.４.５.２　ステージ３ｂ：各セットの相互接続能力の決定

各Ｚチップが各ボックス中の同一のＹチップを接続し、各Ｙチップが各ボードの同一のＸチップを接続しているので、相互接続されたＸ、Ｙ及びＺチップによって一組を形成する。リアライザシステムの好適例においては、６４組を設けている。その各々は、１個のＺチップと、各ボックス中に１個設けている、Ｚチップを用いたＹ—Ｚバスを有する８個のＹチップと、各ボードに１個設けている、各Ｙチップを用いたＸ—Ｙバスを有する６４個のＸチップとを具えている。各々の組の対は、この場合、同一のＸチップを有しているが、このことは許容されている。その理由は、唯１個の組のみを選択し、回路網を相互接続するからである。ＬチップとＸチップのような相互接続されたチップの対の各々を、バスと称する一群のワイヤで接続する。各クロスバー中のバスを、バステーブルにリストする。Ｌ—Ｘバステーブルは、システム全体中の各Ｌ—Ｘクロスバーの各バスに関連するエレメントを有している。各ボックス中の各ボードにはＬ—Ｘクロスバーを設け、各クロスバーには、一組の各Ｌチップ及びＸチップに関係するバスを設ける。このようにして、Ｌ—Ｘバステーブルは５個の寸法を有している。すなわち、ＬＸ〔ボックス〕〔ボード〕〔Ｌチップ〕〔Ｘチップ〕〔バス〕である。同様にして、Ｘ—Ｙバステーブル、すなわちＸＹ〔ボックス〕〔ボード〕〔Ｙチップ〕〔バス〕と、Ｙ—Ｚバステーブル、すなわちＹＺ〔ボックス〕〔Ｚチップ〕〔バス〕とを設ける。テーブル中の各エレメントを、相互接続手続によって、“フリー（free）”又は“ユーズド（used）”とする。ネットリストファイルに送出された入力又は出力Ｉ／Ｏピンがバスを使用する場合、テーブルエレメントを使用する。各組の回路網相互接続能力を、相互接続すべき各バスに対するフリーなバスカウントを捕捉することによって決定する。まず第１番目に、ボックス中のＹチップとＺチップとの間のＹ—Ｚバスについて考える。回路網中の各ボックスにおいて、この組中のＺチップ及びこのボックスのＹチップに関するＹ—Ｚバステーブル中のフリーバスの数をカウントし記憶する。第２番目に、ボード上のＸチップとボックス中のＹチップとの間のＸ—Ｙバスについて考える。すなわち、回路網中の各ボードにおいて、この組のこのボックスのＹチップ及びこのボードのＸチップに関するＸ—Ｙバステーブル中のフリーバスの数をカウントし記憶する。第３番目に、ボード上のＬチップ及びＸチップ間のＬ—Ｘバスについて考える。すなわち、回路網の各ロジックチップにおいて、この組のこのＬチップ及びこのボードのＸチップに関するＬ—Ｘバス中のフリーバスの数をカウントし記憶する。いかなるポイントにおいても、相互接続を完成するのに十分なフリーバスが存在しない場合、この組を故障と認識し、このプロセスを次の組に進める。結果的には、相互接続中の各バスすなわち首尾よく相互接続を達成することのできるクロスバーチップの各組に関するバスカウントを捕捉することとなる。

【００８０】２.４.５.３　ステージ３ｃ：組の選択

多くの組を用いて相互接続することができるので、組の１個を選択し使用するバスのバランスを保持する。このことによって、完全な相互接続機能の開発が保証される。簡単な組選択技術は、全バスカウントが最大である組を選択することである。しかし、このことは局所的な条件を無視している。すべてのレベルにおけるバスカウントの中から、最も大きな最小バスカウントを有する組を選択するのが好ましい。例えば、２組が以下のバスカウントを有しているものと仮定すると、

バス：	ＹＺ	ＹＺ	ＸＹ	ＸＹ	ＬＸ	ＬＸ	ＬＸ
組Ａ：	4	4	4	3	1	3	4
組Ｂ：	3	3	3	3	3	3	3

組Ａは、最大トータル（２３対２１）を有しているが、これを選択するということは、最後に利用することのできる、１個のＬチップ—Ｘチップ対からのＬ—Ｘバスを採用することを意味している。組Ｂは、最も大きな最小（３対１）を有しており、Ｌチップ—Ｘチップ対を閉じることはない。結合の場合、検討の結果、各組から１個の最小を排除し、組を１個選択するまで、最も大きな最小を有している組を選択する。（第１回路網の場合と）実際にすべての組が同一である場合、同一である場合、一つを採用する。これが、使用されている方法である。一組のトライステート回路網について検討を加える場合、特に考慮を要する。一方が階層の上方に向かって加算ゲートにつながっている入力端子のためのバスである。同一の回路網に使用されるこれら２個のバスを、いくつかのチップ対は、有していなければならないため、これらの場合に、選択された組は、少なくとも２個のフリーバスを有していなければならない。バスのトリーノード（すなわち、Ｌ—Ｘバス等のためのＸチップノード）がノンゼロのＤ及びＲカウントと、ノンＮＵＬＬの親とを有している場合、このような場合を検出する。

【００８１】２.４.５.４　ステージ３ｄ：相互接続のネットリスティング

組の選択及びトリー構造を与え、バッファ及び加算ゲートのステートメントをロジック及びクロスバーチップネットリストファイルへ送出することによって、相互接続をネットリストする。このことを、各レベル毎に、まずロジックチップについて行い、その後、X、Y及びZチップについて行う。各チップの相互接続及び方向性を、トリー中のデータを使用することによって決定する。相互接続のバッファ及び回路網に関してのステートメントをネットリストファイルに送出することによって、各接続をネットリストする。（チャイルドチップが存在する場合）チップとチャイルドチップとの接続を、まずネットリストする。各チャイルドチップを順番に検討する。トリーがこのチップを駆動させていることを示している場合、チャイルドチップのドライバを接続しているピンナンバを用い、入力バッファをネットリストする。このチップが1個より多くのドライバを有している場合、別の回路網ネームを各々に用いる。このようにして、これらの回路網ネームは後にネットリストされる加算ゲートによって捕捉される。チャイルドがこのチップを受信していることをツリーが示している場合、チャイルドチップのレシーバを接続しているピンナンバを用いて、出力バッファをネットリストする。このチップ自体がその親からのレシーバである場合、異なる回路網ネームを用い、このチップは親レシーバを接続する。このチップが、そのチャイルド中に1より多くのドライバを具えている場合、加算ゲートをネットリストし、上述したドライバ回路網を接続する。最終的に、（存在するならば）親チップへの接続をネットリストする。チップ又は任意の派生チップがドライバを具えている場合、チップの対及び選択された組に関するバステーブルエントリからドライバに関する相互接続パスを採用するとともに出力バッファをネットリストし、ここで採用したパスを介して親を駆動する。このチップが、ペアレントからのレシーバである場合、バステーブルからパスを選択し、このパスを用いて入力バッファをネットリストする。

【0082】2.4.6　相互接続及びネットリスティング手続についての詳細な規定：

第1の一般的な規定：

回路網に関して4個のクラスを設ける：

シンプルな閉回路網：回路網は1個のドライバを有し、すべての基本要素を同一のLチップ中に設ける。

シンプルなカット回路網：回路網は1個のドライバを有し、基本要素を多数のLチップ中に設ける。

トライステート閉回路網：回路網が1個より多くのドライバを有し、すべての基本要素を同一のLチップ中に設ける。

トライステートカット回路網：回路網が1個より多くのドライバを有し、基本要素を多数のLチップ中に設ける。回路網の‘ソース’とは、その実際の論理値を伝送するチップである：シンプルな回路網では、ソースはドライバを有しているLチップである。トライステート回路網では、ソースはトップモスト（top-most）加算ゲートを有しているチップである。これを決定するために：回路網を走査し、どこに出力ピンを配置しているかを調べる。出力ピンがすべて同一のLチップ上に存在する場合、このLチップがソースである。これ以外の場合で、出力ピンがすべて同一のボード上に存在する場合、このボード上のXチップがソースである。これ以外の場合で、出力ピンがすべて同一のボックス中に存在する場合、このボックス中のYチップがソースである。上記以外の場合、Zチップがソースである。出力ピンのインデックスナンバは、それが、その回路網におけるピンの循環リストにおけるどの出力ピンであるかを示しており、ネットレコードが示すピンから始まり、ゼロから一つずつカウントする。

ステージ1：設計部データ構造中のすべての基本要素を送出する。設計部データ構造中の各Lチップは：Lチップのネットリストファイルがオープンされていない場合には、このLチップのネットリストファイルをオープンする。このLチップの各基本要素は：基本要素ヘッダステートメントをファイルに送出する。この基本要素の各ピンは：（入力設計ファイルからネームを得るための回路網の対象識別子を用いて）接続された回路網のネームを得る。そして、‘N’と称する。

入力ピンの場合：このLチップが回路網のソースを有している場合には、回路網‘N’に接続された入力ピンに関するステートメントを送出する。そうでない場合には、回路網‘N_R’における入力ピンステートメントを送出する。

出力ピンの場合：この出力ピンのインデックスナンバを得て、‘p’と称する。シンプル回路網の場合には回路網‘N’のピンに指示を出す。トライステート閉回路網の場合には回路網‘N_OR_p’のピンに指示を出す。

トライステートカット回路網の場合：これがこのLチップのこの回路網に関する唯一つの出力である場合、回路網‘N’のピンに指示を出す。唯一つの出力ではない場合、回路網‘N—OR—p’のピンに指示を出す。次のピンへ。次の基本要素へ。次のLチップへ。

ステージ2：すべての閉回路網加算ゲートに指示を出す：各トライステート閉回路網は：‘N’と称するこの回路網のネームを得る。このLチップのネットリストファイルがオープンされていない場合には、これをオープンする。‘i’と称する回路網に、何個の出力端子が存在するかをカウントする。‘i’入力ゲートに関するステートメントを送出する：この回路網が‘フローティングハイ’である場合にはNORであり、これ以外の場合にはORであり、（0～i—1のすべてのjについて）回路網‘N_OR_j’に接続された入力端子と、‘N’に接続された出力端子とを有している。次の回路

網へ。
ステージ３：カット回路網と相互接続しているバッファに指示を出すとともに、すべてのカット回路網加算ゲートに指示を出す：すべての相互接続バステーブルのすべてのエレメントを“フリー”にする。各カット回路網（シンプル又はトライステート）は:階層の順番でカット回路網を選択し、第１ボックス回路網等を選択するとともに、この順番の範囲内で最も大きな第一番目の回路網を選択する。
ステージ３Ａ：トリーの構成
回路網の各基本要素は：この基本要素のボックスにトリーノードを設けていない場合には、１を加える。このボックス中の基本要素のボードにトリーノードを設けていない場合には、１を加える。このボックス中のこのボード上のこの基本要素のＬチップにトリーノードを設けていない場合には、１を加える。この基本要素の回路網接続が出力ピン（すなわちドライビィング（driving））である場合には、このＬチップのノードのＤカウントをインクリメントする。上記以外の場合で、このＬチップがこの回路網のソースではない場合、このＬチップのノードのＲカウントをインクリメントする。次の基本要素へ。この回路網のすべての基本要素をトリーで表現する際、唯一つのＸチップノードのみを設けるならば、Ｙチップノードを ＮＵＬＬとする。（すなわち、回路網はボード上に存在している。）
唯一つのＹチップノードのみが存在する場合、Ｚチップノードを ＮＵＬＬとする。（回路網はボックス中に存在する。）
各ノンＮＵＬＬクロスバーレベルにおいて、まずＸチップ、その後Ｙチップ、その後Ｚチップ：
このレベルにおける各ノードは：
Ｄ＝〔Ｄカウントがゼロでないチャイルドノードの数〕
Ｒ＝〔Ｒカウントがゼロでないチャイルドノードの数〕
このノード又は派生をこの回路網のソースとする場合、このノードをＲ＝０にセットする。このノードをソースとし且つ回路網をトライステートにする場合、その“トップサム（top sum）”フラグを真にセットする。次のノードへ。次のレベルへ。
ステージ３Ｂ：各組の相互接続能力の決定
各組は、相互接続すべき各バスのバスカウントを捕捉し、その相互接続能力を決定する。
この組のバスカウントの記憶を割当てる：
Ｙ—Ｚバスカウント：各ボックスの割当て
Ｘ—Ｙバスカウント：各ボードの割当て
Ｌ—Ｘバスカウント：各Ｌチップの割当て
唯一個のボックスのみをこの回路網中に設ける場合：このボックスのナル（ゼロではない）Ｙ—Ｚバスカウントをそのままにしておく。そうでない場合には：各ボックスは：バスアレイ中のフリーバスの数をカウントする。
ＹＺ〔このボックス〕〔この組〕〔バス〕
このボックスのトリーノードがノンＮＵＬＬの親を有し、且つＤ＞０及びＲ＞０である場合、このボックスのバスは“ダブル”である。すなわち、ドライバとレシーバの両方を有している。フリーバスが２よりも少ない場合、この組はこの回路網を接続できない。以上以外の場合で、且つバスが存在しない場合、この組はこの回路網を接続できない。この組が接続できない場合、これを使用不可能とし、次の組に進める。接続できる場合には、トータルを、このボックスのＹ—Ｚバスカウントとしてセーブする。この回路網に、唯一のボードのみを設けている場合、（Ｙ—Ｚ相互接続は必要とされない）：このボードのナル（ゼロでない）Ｘ—Ｙバスカウントをそのままにしておく。２以上のボードを設けている場合：各ボードは：バスアレイなかのフリーバスの数をカウントする。ＸＹ〔このボックス〕〔このボード〕〔このセット〕〔バス〕このバスが“ダブル”であり且つバスが２より少ない場合、又は、バスを設けていない場合、この組はこの回路網を接続できない。この組を使用不可能とし、次の組に進む。上記でない場合：トータルを、このボードのＸ—Ｙバスカウントとしてセーブする。各Ｌチップは：バスアレイ中のフリーバスの数をカウントする。ＬＸ〔このボックス〕〔このボード〕〔このＬチップ〕〔この組〕〔バス〕このバスが“ダブル”であり、且つバスが２より少ない場合、又は、バスを設けていない場合、この組は、この回路網を接続できない：この組を、使用不可能とし、次の組に進む。上記でない場合には、トータルを、このＬチップのＬ—Ｘバスカウントとしてセーブする。このボードの次のＬチップへ。このボックスの次のボードへ。次のボックスへ次の組へ
ステージ３Ｃ：回路網の選択：回路網を接続することのできる各組は：この組のすべてのバスカウントの中から最小バスカウントを見出す。次の組へ。これらの最小バスカウントの中から最大のものを見出す。検討の後、最大の最小バスカウントよりも少ないバスカウントを有するすべての組を除去する。組が存在しない場合には、この回路網を相互接続することができない。一組だけが残る場合には、この回路網の組を選択する。２以上の組が残る場合には、すべての最小バスカウントの中から次の最も大きい最小を見出す。検討の後、これよりも小さいバスガウントの組すべてを除去する。一組が残るまで、又は、すべての残っている組のバスカウントが同一となるまで、これを繰り返す。この回路網の残っている組の内の任意の１個を選択する。すべての組のすべてのバスカウントの記憶を解除する。
ステージ３Ｄ：相互接続のネットリスト以下で用いられる手続の規定：ドライバ（又はレシーバ）バスを得る又は確保するために：
１）このレベルのバステーブル中のフリーエレメント、このノードのチップナンバ及びペアレントノードのチップナンバから、バスを選択する。

２）使用されたパスのテーブルエレメントをマークする。

３）このノードのドライバ（又はレシーバ）パスナンバエントリの中のパスナンバとして、どのパスを用いたかを記憶する。Ｉ／Ｏピンナンバを導出するために：

１）２個のノードのチップナンバ及び組ナンバから、このノードのチップとチャイルドノードのチップ（又は、場合によっては親ノードのチップ）との同一性を確認する。これによって、含まれている特定のパス（例えば、Ｌ４－Ｘ５，又はBoard３－Ｙ７のようなパス）を識別する。

２）パスナンバがチップの対を接続している幾つかのパスの内の一つのパスを示しているということをリコールする。チップ、パス及びパスナンバを与え、Ｉ／Ｏピンナンバ情報を有している索引テーブルから、このパスを接続しているピンナンバを読出す。パスを用いて、バッファ（入出力）に指示を出すために：

１）パスナンバをこのノードから得る。又は、チャイルドパスが特定される場合には、パスナンバをそのチャイルドノードから得る。ドライバ又はレシーバパスナンバを、指示されたようにして得る。

２）パスナンバを用いて、このバッファのＩ／Ｏピンナンバを得る。

３）入力バッファであるか出力バッファであるかに応じて、このノードチップのネットリストファイルに、基本要素ステートメントを送出する。この際、指示されるように、入出力回路網ネームを用いるとともに、そのＩ／Ｏピンに対して得られたピンナンバを用いる。

相互接続のネットリスト手続：‘Ｎ’と称するこの回路網のネームを得る。各ノンＮＵＬＬレベルの第１Ｌチップ、その後Ｘチップ、Ｙチップ及びＺチップにおいて：トリー全体におけるこのレベルにおける各ノードは：準備していない場合には、このノードのチップに関するネットリストファイルをオープンする。レベルがＸ、Ｙ又はＺである場合：このノードの下位の各チャイルドノードは：カウンタ‘ｉ’をゼロにセットする。チャイルドのＤ＞０である場合：（チャイルドとはドライバである）このノードのＤ＝１である場合：‘Ｎ＿Ｄ＿ｃ‘から‘Ｎ’へと入力バッファに指示を出す。（ここで‘ｃ’は、チャイルドノードのナンバである。）この際チャイルドドライバパスを使用している。このノードのＤ＞１である場合：‘Ｎ＿Ｄ＿ｃ’から‘Ｎ＿ＯＲ＿ｉ’へと入力バッファへと指示を出す。この際、チャイルドのドライバスを使用し、‘ｉ’をインクリメントする。チャイルドのＲ＞０である場合：（チャイルドはレシーバである）このノードのＤ＞０、且つこのノードのＲ＝０の場合：‘Ｎ’から‘Ｎ＿Ｒ＿ｃ’へと出力バッファに指示を出す。この際、チャイルドシートのレシーバパスを使用する。そうでない場合：‘Ｎ＿Ｒ’から‘Ｎ＿Ｒ＿ｃ’へと出力バッファに指示を出す。この際、チャイルドのレシーバパスを使用する。次のチャイルドノードへ。このノードのＤ＞１である場合：（ノードは加算ゲートを有している）‘ｉ’入力ゲートに指示を出す：この回路網が‘フローティングハイ’であり、且つこのノードの‘トップサム’フラグが真である場合、ＮＯＲである。そうでない場合には、ＯＲである。この際、（０～ｉ－１のすべてのｊに対する）‘Ｎ＿ＯＲ＿ｊ’に接続された入力端子と、‘Ｎ’に接続された出力端子とを有している。このノードのＤ＞０で且つこのノードがノンＮＵＬＬ親を有している場合：（ノードはドライバである）ドライバパスを確保するとともに、受信する。‘Ｎ’から‘Ｎ＿Ｄ’へと出力バッファに指示を出す。この際、ドライバパスを使用する。このノードのＲ＞０である場合：（ノードはレシーバである）レシーバパスを得て確保するとともに、受信する。‘Ｎ＿Ｐ’から‘Ｎ＿Ｒ’へと入力バッファへと指示を出す。この際、受信パスを使用する。このレベルの次のノードへ。次のレベルへ。次のカット回路網へ。すべてのオープンネットリストファイルをクローズする。

【００８３】２.４.７　２例の回路網

図４７ａは、‘ＢＸ’と称する、シンプル回路網のオリジナル入力設計部を示している。これは、１個のドライバと３個のレシーバとを具え、同一ボックス中の一方のボード上の２個のロジックチップ及び他方のボード上の１個のロジックチップに及んでいる。この回路網のステージ３ａによって構成された相互接続トリーを、図４７ｂに示す。どのようにして、各ロジックチップ、各ボードに対するノード及びボックスに対するノードを設けるかに注意しなければならない。ロジックチップノードは、特定のロジックチップに対応している。ボードノードは各ボード上のＸチップに対応しており、ボックスノードはＹチップに対応している。Ｚチップは、この回路網では必要ない。正確には、どのＸ及びＹチップを使用するかはどの組を選択するかに依存しており、これは、トリー中には示されていない。Ｄ及びＲの値を各ノード毎に示している。ノードがレシーバであっても、ＬＯがＤ＝０であることに注意する。その理由は、ノードがこの回路網のソースノードであり、他のノードとは異なりソースノードから値を受信する必要がないからである。ボード２のノードにおいては、そのＲカウントは初期値が１であり、Ｌ４のレシーバをカウントする。ソースが派生であるため、Ｒカウンドがゼロにセットされていたことを示している。送出された回路網ネームは、これらの回路網を用いて示されている。実際の相互接続の構造によって、トリーの構造及び各ノードのＤ及びＲカウントをどのようにして表わすかを述べる。図４８ａは、‘ＥＸ’と称するトライステート回路網のオリジナル入力設計部を示している。これは、同一ボックス中の一方のボード上の２個のロジックチップ及び他方のボード上の１個のロジックチップに及んでいる３個のトライステ

ートドライバと、２個のボックスにおける３個のボード上の４個のＬチップに及んでいる６個のレシーバとを具えている。この回路網のステージ３ａによって構成される相互接続ツリーを、図４８ｂに示す。この回路網はボックスに及んでいるため、Ｚレベルクロスバーを用いる。２個のトライステートドライバを有しているために、ボード２のノードはＤ＝２であることに注意しなければならない。Ｘチップは、加算ゲートを具え、ボード２のＬチップからのタームを捕捉する。回路網のソースであるボックス２のノードも同様であり、これを、“トップサム”とする。このＹチップは、トップモスト加算ゲートを具え、ボード２及び３からのタームを捕捉する。ボックス２のノード及びそのＺペアレントノードはソースを具え、これらのＲカウントをゼロにする。各ロジックチップ及びクロスバーチップに関するネットリストファイルに送出される実際のゲート及びバッファと、いかにして、これらを相互接続するかを、図４９に示す。設計変換によって、各トライステートドライバをどのようにＡＮＤゲートに変換するかを注意しなければならない。これらの出力を、Ｘ及びＹレベルの加算ゲートによって捕捉する。受信入力は、“トップサム”ノード、すなわちボックス２のＹチップから伝送される。ボックス２のレシーバは、相互接続へつながっているバスによって駆動される。ボックス６のレシーバは、Ｚレベルクロスバーチップを介して駆動される。

【００８４】３　リアライザシステムの応用

３．１　リアライザロジックシミュレーションシステム

ロジックシミュレータは、ハードウェア又はソフトウェアによって実現されるシステムである。このシステムは、入力設計、一組の設計部への刺激、及び或る期間中の刺激の方向を受信するとともに、一組の刺激を出力する。この刺激によって、実際の入力設計部を実現し、所定の同一の刺激を発生させることを予測する。刺激及び応答は、特定の時間に、特定の設計回路網のロジック状態を伝送するものである。シミュレータユーザが、入力設計ファイルの形態で設計部の記述のみを供給するということが重要な特性であり、短期間に設計部を変更するとともに、これに再度刺激を与えることができる。現在のソストウェアロジックシミュレータ設計部の演算は、コンピュータソフトウェアプログラムを用いており、設計部のオペレーションを予測するシーケンシャルなアルゴリズムを実行する（“An Introduction to Degital Simulation”,Mentor Graphics Corp.,Beaverton,Oregon,1989）。よく知られているように、イベントドライブされたコードアルゴリズム、又は、コンパイルされたコードアルゴリズムのいずれか一方を使用する。現在のハードウェアロジックシミュレータ設計部の演算とは、ソフトウェアシミュレータに使用されるのと同一の、イベントドライブされたコードアルゴリズム、又は、コンパイルされたコードシーケンシャルアルゴリズムを実行するハードウェアを構成することである。アルゴリズムの並列処理を開発及び／又は特別なアルゴリズムオペレーションを直接実現することで、ハードウェアはその実行による利益を得ることができる。このことは、一般的な目的のコンピュータ実行ソフトウェアにおいては不可能である。現在のハードウェアロジックシミュレータは、入力設計部の応答を予測するシーケンシャルなアルゴリズムを実行することで動作する。ロジックシミュレータを構成する新しい手段は、リアライザシステムに基づいている。リアライザロジックシミュレータシステムは、入力設計を受信し、これをリアライザハードウェアのロジック及び相互接続チップの構成に変換する。この際、リアライザ設計変換システムを使用する。リアライザロジックシミュレータシステムは、一組の設計部への刺激と、ある期間のシミュレートする方向とを受信し、ベクトルメモリを介し、実現される設計部に刺激を与え、ベクトルメモリを介して、実現される設計部からの一組の応答を捕捉する。応答は、入力設計部を実際に実現することによって、所定の同一の刺激を発生させることに対応している。その理由は、前記刺激に対応させて、設計部をハードウェアによって実際に実現するからである。このことは、現在のロジックシミュレーションシステムのすべてが、設計部の刺激に対する応答を予測するシーケンシャルアルゴリズムを実行するが、リアライザロジックシミュレータは実際の設計部の実現を行い、設計部の刺激に対する応答を決定するという点において、すべての現行のロジックシミュレーションシステムとは異なる。主な利点は、実現された設計部が、シーケンシャルアルゴリズムが応答を予測できるより速く、種々の速さで応答を発生させるということである。リアライザロジックシミュレーションシステムは、（すでに説明した）リアライザ設計変換システムと、ロジックシミュレータ刺激及び応答伝送システムと、リアライザハードウェアシステム及びホストコンピュータと相俟って、カーネルを作動させるロジックシミュレータとから成っている（図５０）。

【００８５】３．１．１　ロジックシミュレーション刺激及び応答の変換システム

このシステムは、ユーザが作成した刺激イベント入力ファイルを、直接ベクトルメモリにロードすることのできる刺激データを含むバイナリーファイルに変換するとともに、ベクトルメモリから読出されるバイナリー応答データを有するファイルから、ユーザが読出し可能な応答イベント出力ファイルへ、応答を変換する。刺激及び応答イベントは、回路網ネーム、時間及び新しい回路網の状態値から成っている。変換とは、回路網ネームとベクトルメモリビットとの間の変換、及び、シミュレーションの‘実時間’とベクトルメモリロケーションとの間の変換である。時間変換は、刺激イベントを有する各特定の時間をベクトルメモリロケーションに印すとともに、

このベクトルメモリロケーションにおける応答イベントをこの時刻に発生したものとして報告する。好適例では、刺激入力イベントファイル及び応答出力イベントファイルを、Mentor Graphics Logfiles（“Quick Sim Family Reference Manual”, MentorGraphics Corp.,Beaverton,Oregon,1989）としている。これは、一連の時刻、回路網ネーム及び、新たな回路網状態値を含むテキストファイルである。ＥＤＡシステム中のバッチシミュレーションインタフェースツールによって、刺激入力イベントファイルを作成するとともに、応答出力イベントファイルを解釈する。好適例では、このツールを、Mentor Graphics のRSIMツールとする。ここでは、このセクションで後に説明するように、すべての基本要素を、ゼロディレイでシミュレートするものと仮定する。刺激イベント入力ファイルを、刺激バイナリーファイルへ変換するためには：

1）刺激入力イベントファイルを読出す。時間の経過に応じて、刺激イベントをオーダするとともに、何個の異なる時刻がイベントを有するかを決定する。

2）設計変換システムが出力するこの設計部中の各ベクトルメモリに対する対応テーブルを読出す。

3）各ベクトルメモリロケーションは、１以上の刺激イベントを有する時刻に対応している。各々の異なる刺激イベント時刻に、十分なベクトルメモリロケーションが存在しない場合、ステップ５及び６を必要なだけリピートし、すべてのこのような時刻に十分な刺激バイナリーファイルを出力する。このファイルは、各々メモリに適合する刺激を有している。

4）ベクトルアレイ“Ｖ０”，“Ｖ１”等の記憶を割当てる。その各々は、ロケーションのナンバ及び回路網幅と一致しており、シミュレートされる設計部に用いられるベクトルメモリを用いている。ベクトルアレイと同じ長さを有するタイムアレイ“Ｔ”の記憶を割当てる。“ラストベクトル”バッファ“Ｂ０”、“Ｂ１”等を割当てる。このバッファは、各ベクトルメモリに対するものであり、各々その回路網と同じ幅であり、これらをゼロに初期化する。

5）ベクトルアレイインデックスカウンタ‘ｖ’を、ゼロにセットする。１以上の刺激イベントを有する各時刻のうち、最も早い第１番目の時刻において、ベクトルメモリ‘ｎ’及び、この回路網のベクトルメモリビットポジション‘ｉ’を設定する。この際、このイベントの回路網の対応テーブルエントリを使用する。このイベントに対する新しい値を、Ｖｎ[ｖ] ビットｉ及びＢｎビットｉに書込む。次のイベントへ。Ｖ０[ｖ]、Ｖ１[ｖ] 等の内容の各々を、Ｂ０、Ｂ１等に書込む。Ｔ[ｖ]中のこの時刻を記憶する。ｖをインクリメントする。刺激イベントを有する次の時刻へ。

6）ベクトルアレイＶ０、Ｖ１等、タイムアレイＴ及びサイクルカウント‘ｖ’を刺激バイナリーファイルに書込む。応答バイナリーファイルを、応答イベント出力ファイルに変換するために：

1）ベクトルアレイＶ０、Ｖ１等、タイムアレイＴ及びサイクルカウント‘ｖ’を応答バイナリーファイルから読出す。各ベクトルメモリロケーションは、１以上の刺激イベントを有する時刻と一致している。各々異なる刺激イベント時刻に対して、ベクトルメモリロケーションが十分でない場合、ステップ１～４を必要なだけ繰り返し、すべての応答バイナリーファイルをこれらのアレイ中に読出す。

2）設計部変換システムが出力する、この設計部中の各ベクトルメモリに対する対応テーブルを読出す。

3）“ラストベクトル”バッファ“Ｂ０”、“Ｂ１”等を割当てる。これは、各々のベクトルメモリに対するものであり、各々その回路網と同じ幅を有しており、これをゼロに初期化する。

4）ベクトルアレイインデックスカウンタ‘ｖ’をゼロにセットする。ベクトルアレイ中の各ロケーションは：Ｖ０[ｖ]をＢ０と比較し、Ｖ１[ｖ]をＢ１と比較する。Ｖｎ[ｖ]のビットとＢｎとの各々の差によって：このビットのベクトルメモリ及びベクトルメモリビットポジションに対応する回路網ネームを配置する。この際、このメモリに対する対応テーブルを使用する。新しい応答イベントを出力ファイルに書込む。この際、回路網ネーム、新しいビットの値及び時刻Ｔ[ｖ]を用いる。次のイベントへＶ０[ｖ]、Ｖ１[ｖ]等の内容の各々を、Ｂ０、Ｂ１等に書込む。ｖをインクリメントする。次のロケーションへ。

【００８６】３.１.２　ロジックシミュレーションオペレーティングカーネルオペレーティングカーネルは、シミュレートされる設計部のリアライザシステムを構成し、刺激を与えるとともに応答を捕捉する。このことは、ホストコンピュータが行う。各セクションに説明されているように、オペレーティングカーネルはロジックチップ及び相互接続チップを構成し、ベクトルメモリ及び設計メモリを読出し及び書込むとともに、ホストインタフェースを介してクロック発生器及びリセット発生器を制御する。シミュレーションを実行するために：

1）設計部の構成ファイルを読出すとともに、これを用いて、構成のセクションで説明したように、すべてのリアライザロジックチップ及び相互接続チップを構成する。初期設計メモリデータをファイルから読出すとともに、これを設計部メモリに書込む。

2）刺激バイナリファイルを読出す。ホストインタフェースを介して、対応するベクトルメモリ中にベクトルアレイの内容を記憶する。

3）ベクトルメモリモジュール中のすべてのベクトルメモリの内容をクリアする。設計部リセット発生器を周期化し、実現される設計部を初期化する。

4）‘ｖ’周期のＥＣＬＫ回路網のクロック発生器をイ

ネーブルする。このことによって、ベクトルメモリがこれらの刺激データを送出でき、この刺激に従って、実現される設計部を作動させるとともに、ベクトルメモリが応答データを捕捉する。このことについては、刺激／応答のセクションにおいて説明されている。

５）ベクトルメモリの内容を読出すとともに、これらをタイムアレイ“Ｔ”及びサイクルカウント“ｖ”とともに、応答バイナリーファイルに記憶する。

６）ベクトルメモリの容量が不十分であるために１より多くの刺激バイナリーファイルを設ける場合、各ファイル毎にステップ２～５を繰り返す。

７）ユーザ試験のためのファイル中の設計メモリ内容をセーブする。

【００８７】３．１．３　リアライザロジックシミュレーションシステムの使用

リアライザロジックシミュレータを用いて、入力設計部をシミュレートするために：

１）ベクトルメモリ接続を示している特性を用いて、刺激すべき回路網とこの応答を捕捉するための回路網とをマークすることによって、ＥＤＡシステムの設計作成ツールを使用し、入力設計部を準備する。必要ならば、初期設計部メモリデータファイルを準備する。ＥＤＡシステムのバッチシミュレーションインタフェースツールを用いて、刺激イベント入力ファイルを準備する。

２）リアライザ設計変換システムを用いて入力設計部を変換し、構成ファイル及びベクトルメモリ回路網対応テーブルファイルを出力する。

３）刺激及び応答変換システムをランさせ、刺激イベント入力ファイルを刺激バイナリーファイルに変換する。

４）オペレーティングカーネルをランさせ、シミュレーションを行うとともに、応答バイナリーファイルを出力する。

５）刺激及び応答変換システムをランさせ、応答バイナリーファイルを応答イベント出力ファイルに変換する。

６）ＥＤＡシステムのバッチシミュレーションインタフェースツールを用い、応答イベント出力ファイルを翻訳する。

７）入力設計部、初期設計部メモリファイル及び／又は刺激イベント入力ファイルをシミュレーションの結果によって示されるように変更し、必要ならば、ステップ２）～６）をリピートする。リアライザロジックシミュレーションシステムの会話形の変形例では、刺激に対してはスティミュレータを用い、応答に対してはサンプラを用いている。構成及びオペレーションは以下を除き同様のものである。すなわち、バッチシミュレーションインタフェースツールの代わりに会話形シミュレーションインタフェースツールを用いて、ファイルを介する代わりに刺激及び応答変換システムと直接通信しており、又、同時に作動する会話形シミュレーションインタフェースツールオペレーティングカーネルを用いて、刺激及び応答変換システムが、ファイルを介する代わりに直接オペレーティングカーネルと通信している。イベントを有する各タイムステップを、ベクトルメモリロケーションではなく、エッジ検知タイプのシミュレータに作成する。

【００８８】３．１．４　３以上の論理状態の実現

リアライザシステムにおいて、２個のロジック状態を実現するのは実用的なことである：高ロジック状態（Ｈ）すなわち真と、低ロジック状態（Ｌ）すなわち偽であり、リアライザシステムの単一の信号を用いて、入力設計部中に直接各回路網を実現することによって実現される。ロジックシミュレーション環境において、時には３以上のロジック信号の状態を表現することが望まれる。例えば、第３の状態、“未知の（Ｘ）”を用いて、初期化されていないロジック変数又はあいまいなロジック状態を表現する。高インピーダンス状態（Ｚ）は、トライステートバスのようなワイヤ結合のバスを実現するのに役立つ。リアライザシステムの幾つかの例において、高インピーダンス状態を直接実現することができる。例えば設計部にトライステート回路網が必要とされる場合、ロジックチップ及び任意の必要な相互接続がトライステートバス機能を構成する能力を有している限り、リアライザシステムのトライステートバスによって設計部が実現される。代わりに、１個の回路網を２以上の信号へ、以下のように符号化することによって任意のロジック状態を実現する：実現すべき状態の数を決定する。すべての状態を単一に符号化するのに必要とされる最小バイナリービット数を決定し、これを‘ｎ’と称する。‘ｎ’個の実際のバイナリーロジック信号によって設計部中の回路網を実現する。例えば、３個の状態（Ｈ，Ｌ，Ｘ）を必要とする場合、２個の実際のバイナリー信号を用いてリアライザシステム中の単一の設計回路網を実現する。基本要素変換ステージの間にこの変換を行い、これらの新しいバイナリー信号を設計部データ構造に入力し、オリジナル設計回路網を置換する。更に、設計部中のロジック基本要素を、多数状態ロジック機能に従って動作するロジック回路網で実現する。例えば３個の状態（Ｈ＝高ロジック状態，Ｌ＝低ロジック状態，Ｘ＝未知）を用いる場合、設計部中の２入力ＡＮＤゲートを、３状態ＡＮＤ機能に従って動作するロジック回路に従って実現する（図５１ａ）。いずれかの入力端子をＸとし、入力端子が存在しないことをＬとする場合、出力端子に生じるＸ状態を用いて、３状態シミュレータの如く、ロジック機能が動作する（図５１ｂ）。この回路網は、２個の２ビット入力端子と、１個の２ビット出力端子とを有している（図５１ｃ）。マルチステートを実現するこの技術を、設計解析のために必要とされるように入力設計部の全体又は設計部の一部分にのみ用いることができる。２個より多くの状態でシミュレートされる回路網をこのようにして入力設計ファイル中に作成し、設

計部リーダは設計部データ構造中のこのことに注目し、基本要素コンバータは基本要素に対する上記の代用回路網及び基本要素に対する多数の回路網を作成する。ロジック基本要素が、２状態の回路網接続と３以上の状態の回路網との混合を有している場合、回路網の条件に従って作動するロジック回路網を使用する。もしそうでなければ、上述したシミュレーション動作となる。

【００８９】３.１.５　リアライザのディレイの表現

現在のロジックシミュレータでは、種々の方法で、信号がロジックエレメントを通過する際のタイムディレイを作る。リアライザのロジックチップ中のロジックは実際のハードウェアであるため、そのディレイ特性は完全に正確に規定することはできず、ロジックディレイを直接作成することはできない。ロジックディレイは、プログラムを実行するシミュレータ中の特別の方法を用いることによって、及び／又は設計変換プロセスの間、ディレイを形成する特別のロジック機能を挿入することによって作成する。ゼロディレイ、ユニットディレイ又はリアルディレイとして、実現するシミュレーション中にディレイを形成することができる。この選択はユーザによって成され、これをリアライザロジックシミュレータシステムに指定する。

【００９０】３.１.５.１　ゼロディレイ

ゼロディレイとは、リアルタイムディレイを形成することなく、ディレイをゼロとして処理し、シミュレーションを行うことをいう。例えば、刺激イベントが、時刻‘ｉ’に、結合ロジックのみを介して出力端子と接続されている入力端子に生じる場合、この出力端子の応答イベントは、時刻‘ｔ’に生じるものとして報告される。ゼロディレイのために、設計部変換システムは特別なロジック機能を挿入しない。上記の如く、メインリアライザロジックシミュレーションシステムにおいて説明した方法に従って、シミュレーションを行う。

【００９１】３.１.５.２　ディレイ依存機能

設計部に任意の遅延依存機能を設けると複雑なものとなる。ゼロ遅延タイミングモデルではここまでには至らない。クローズドループ機能、すなわちクロスカプルドゲートのように、非同期フィードバックを設ける場合、無条件に記憶装置を設ける。記憶機能は相対的なディレイに依存している。ディレイ依存機能は、ディレイをオープンループ機能に用いると他の形態のものとなる。この一例としては入力端子で接続されたディレイ素子を有するイクスクルーシブＯＲゲートがある（図５２ａ）。イクスクルーシブＯＲゲートの出力は、信号がディレイ素子を介して伝播するのに必要とされる時間においてはハイである。この回路網に供給される信号が変化すると出力端子にパルスを出力する（図５２ｂ）。実際のリアライザロジックディレイはゼロではないため、これを直接コントロールできない場合、クロスカプルドゲートのような多くのクローズドループ及び幾つかのオープンループの場合、ディレイ依存機能は正しく動作する。しかし、ユーザは確実に実現された設計部が意図するように動作することを要求する。現在のタイミング分析ツールは、非同期フィードバックの瞬間を自動的に見出し、レポートするとともに、オープンループディレイに依存する行動を検出する。リアライザ設計変換システムは、ユーザが必要とするならば、タイミング分析ツールを用いることによってタイミング分析を行う。好適実現例では、Mentor Graphics Quick Pathタイミング分析ツールを用いる（“Quick Path User's Manual”, Mentor Graphics Corp., Beaverton, Oregon, 1989）。

１）設計変換プロセスの一部として、ＥＲＣＧＡネットリスト変換ツールは、内部相互接続及びロジック遅延の評価を出力する。これらは、レポートファイルへ送出される。

２）すべてのネットリストを変換した後、データをリポートファイルから読出し、設計部データ構造中に入力する。この際、基本要素又は回路網と関連する各ディレイ評価を用いる。

３）設計部データ構造を設計ファイルに書込む。

４）タイミング分析ツールを設計ファイルに適用する。タイミングアナライザによって検出される任意の起こり得る変化をユーザに報告する。ユーザは、入力設計ファイルを適切に評価し、修正する。

【００９２】３.１.５.３．ユニットディレイ

ユニットディレイモデルとは、各ロジック基本要素が１単位（ユニット）のディレイを有するように形成したものである。ディレイ依存性を用いて、このような形成を、時々設計部に使用し、正しい動作を保証する。適切な特性を入力設計ファイル中の基本要素に加えることによって、ユーザはゼロディレイ基本要素と合成されたユニットディレイ基本要素を指定する。自動的に、各ユニットディレイロジックエレメントの出力端子にフリップフロップを設け、ユニットディレイを形成する。これらのフリップフロップを共通クロックに接続し、この共通クロックは、第２クロック発生器によってシミュレーションの各単位時間毎に一回の周期を成す。これらのフリップフロップ及び‘タイムクロック’回路網を、基本要素変換プロセスによって設計データ構造に加える。ユニットディレイを用いてシミュレートされるロジック設計回路網の一例としては、クロスカプルドゲートを用いて作成されたフリップフロップがある（図５３ａ）。各ゲートを、その出力端子にユニットディレイフリップフロップを設けて構成する（図５３ｂ）。連続的なタイムクロック及び入力信号を与える最終的なオペレーションは、ユニットディレイゲートを有するフリップフロップのオペレーションである（図５３ｃ）。ユニットディレイシミュレーションのためのリアライザロジックシミュレータは、以下の変更を伴うものの、ゼロディレイと同様の方法を用いて動作する。

・ユーザは、どの位の時間が１ユニットに相当するのかを指示する。
・刺激及び応答回数を、ユーザが指定する時間単位の倍数‘M’に制限する。
・各ベクトルメモリロケーションは、M時間単位に対応しており、この時の刺激イベントが存在しているかどうかとは無関係である。
・刺激及び応答変換システムは、これらの仕様を用い、これらの対応関係に従って、イベントとベクトルメモリロケーションとの間にマップを作成する。
・最終的に、刺激イベントを有していない時刻は、前のロケーションと同一の内容を有するベクトルメモリロケーションによって表現される。
・オペレーティングカーネルは、‘タイムクロック’クロック発生器の周波数をＥＣＬＫの周波数のM倍にセットし、互いに同期をとりながら動作するように指示する。オペレーションの間、各M時間単位毎に１個のＥＣＬＫ、従って一組の刺激及び応答が存在する。

【００９３】3.1.5.4 リアルディレイ
リアルディレイ、すなわち種々の時間単位によるディレイを、ロジックチップ中の特定のハードウェア構成を用いて実現する。このハードウェア構成は、設計変換の間、各リアルディレイロジックエレメントの設計部データ構造に自動的に挿入される。これには幾つかの技術がある：各ロジック基本要素出力端子と直列にシリアルシフトレジスタを構成する。その長さを、各ケースに必要とされるディレイ単位の数に対応させて構成する。すべてのシフトレジスタを、各時間単位に対して１回の周期をなすように、共通‘タイムクロック’でクロックする。このようにして、シフトレジスタは‘n’ユニット（単位）リアルディレイとして作用する。ここで‘n’はレジスタの長さである（図５４ａ、ディレイレジスタ中の値に従って、マルチプレクサを介して選択される）。代わりに、有限状態マシン（ＦＳＭ）と、１以上のスターティングカウント（starting count）に対する記憶装置を有するカウンタとを、各ロジック基本要素出力端子と直列に構成する（図５４ｂ）。ＦＳＭは、ロジック基本要素出力状態の変換を検出する。各状態の変換において、発生した特定種類の状態変換（上昇（rising）又は下降（falling））にとって適切なスターティングカウントを用いて、カウンタはＦＳＭによってロードされる。すべてのカウンタは、各時間単位毎に１回の周期を成すように、共通‘タイムクロック’を周期化する。カウントがゼロになった場合、ＦＳＭは、出力状態変換を、ディレイされた出力へと送り、その接続された入力端子へと伝播する（図５４ｃ参照）。

【００９４】3.1.6 リアライザシミュレータから他のシミュレータへの状態の伝送
リアライザロジックシミュレータシステムは極めて高速であるという利点を有し、このため、ソフトウェア又は他のイベントドライブによるシミュレータよりも多くの種々のテストサイクルで処理することができる。このシステムは、ディレイ及び他の時間に関連する項目を表示できず、且つ、設計部中のすべてのノードを監視できないという不利な点も有している。慣用のイベントドライブによるソフトウェアシミュレータはかなり低速であるが、項目の表現及び、刺激及び監視のためのすべての回路網ノードへのアクセスができるという利点を有する。しかし、慣用のイベントドライブによるソフトウェアシミュレータは、非常に低速であるため、シミュレートされた設計部を、初期状態から何１００万又は何１０億周期も離れた誤った状態に送るということは実際には生じない。誤った状態は、実際に、起こり得ないということがわかる。初期状態、すなわち、内部フリップフロップ及びロジックゲート出力の値を読出すことができる(Xilinx ＬＣＡのような）ロジックチップを用いてリアライザシステムを構成する場合、実現されるシミュレーションはストップされ、設計部全体の状態が読出される。リアライザロジックシミュレータと他のシミュレータとを結合させることによって、シミュレートされた設計部の状態（すなわち、設計部中のすべての内部記憶装置の値）が、一方から他方へと伝達される。この際、以下の方法に従っている：
1）同一の設計部を、両方のシミュレータにロードする。
2）リアライザロジックシミュレータ中の設計部は、数サイクルの間、以下のようにシミュレートされる。すなわち、詳細に監視されるべきエラー又は他の条件の発生前の状態へ短時間で設計部を変換する。
3）この時、リアライザ刺激クロックはストップされ、設計部の全状態がロジックチップから読出される。
4）この時、他のシミュレータで表現されている設計部を初期化し、リアライザに基づくシミュレータから読出される状態に適合させる。
5）シミュレーションを、他のシミュレータに進める。
このようにして、リアライザロジックシミュレータの究極的な速度を用いて、長すぎるために、他の方法で取り除くことのできないエラーを除去し、他のシミュレータの詳細及び可視性を用いてエラーの原因を分析することができる。

【００９５】3.2 リアライザフォールトシミュレーションシステム
フォールトシミュレーションとは、テストベクトルを開発、修正するのに用いられるロジックシミュレーションの変形、すなわち、組立て後、設計部、一般的には集積回路の正確性をテストするのに用いられる刺激の組である。ユーザ設計によるフォールティバージョン（faulty version）をテストベクトル刺激を用いてシミュレートするとともに、グッド（good）バージョンと比較し、テストベクトル刺激がグッドバージョンの応答とは別の

応答を発生させるかどうかを調べる。別の応答を発生させるならば、テストベクトル刺激が故障（フォールト）を検出していることを示している。故障の多いセットに対しては、このことが繰返される。このことは、できる限り多くの故障を検出する一組のテストベクトルを開発することを目的としている。一般的に、２個の故障を入力設計部の各回路網においてシミュレートする。すなわち、回路網が“スタックアットロー（stack-at-low）”と称する、常にロー状態である場合と、“スタックアットハイ（stack-at-high）”と称する、常にハイ状態である場合とがある。一般的に、入力設計部が何千個もの回路網及びテストベクトルを有し、且つ、フォールトシミュレーションが各新しいテストベクトルのバージョン毎に繰り返されるため、このことは、極めて時間のかかるタスクである。フォールトシミュレーションを構成する新しい手段は、リアライザシステムに基づいている。リアライザロジックシミュレータの方法を、フォールトシミュレーションの修正に関して用いる。シリアルフォールトシミュレーション技術(“Quick Sim Family Reference Manual”, Mentor Graphics Corp., Beaverton, Oregon, 1989)を用いる：各故障に関しては：

１）実現された設計部を修正し、故障を伝える。

２）刺激を与え、設計部を作動させ、応答を良好設計の応答と比較し、相異をフラグで合図する。

３）故障を取除き、この故障による相異が存在しているかどうかを記録する。

現行のフォールトシミュレーションシステムが、故障設計部の刺激に対する応答を予測するシーケンシャルなアルゴリズムを実行するのに対して、リアライザフォールトシミュレーションでは、実際の故障設計部を実現させ、設計部の刺激に対する応答を決定するという点において、両者は相違する。主な利点は、実現される設計部が、シーケンシャルなアルゴリズムが応答できるよりも速い種々の速度で応答を発することである。リアライザロジック及び相互接続チップで構成したように、故障は、直接設計部に伝えられる。故障を入力設計回路網に伝えるために：入力設計部の回路網が、ロジックチップ中に対応回路網を有している場合：フォールト構成を用いて、回路網に接続された各ロジックチップを再構成する。これは、回路網に接続された入力端子を、故障に従って、一定のハイ又はローに接続している点を除き、オリジナルの構成と同一である。入力設計部中の回路網が、ロジックチップ中に対応する回路網を有していない場合：対応する回路網はロジックチップのロジック機能に包摂されており、ロジックチップをフォールト構成を用いて再構成する。これは、回路網に包摂されるロジック機能を、回路網が故障に応じて、常にハイ又はローとなるように作動する構成にする点を除き、オリジナルの構成と同一である。故障を取り除くために、オリジナルの構成を用いて、チップを再構成する。リアライザフォールトシミュレータは、以下の相異点を有するものの、リアライザロジックシミュレータと本質的に同様のものである（図５５）：リアライザフォールトシミュレータとは、フォールトコンフィギュレータ(configurator)であり、ロジックシミュレータの上位の設計部変換システムの付加的部分を構成している。リアライザフォールトシミュレータは、以下のような各々の故障に対して構成ファイルの相違を出力する：

１）一時的に、故障を設計部データ構造に伝える。

２）どのロジックチップが故障による設計部の変化によって影響されるかを決定する。

３）影響を受けたロジックチップのネットリストファイルを送出する。

４）ＥＲＣＧＡネットリスト変換ツールを用いて、影響を受けたロジックチップの構成ファイルを出力する。

５）フォールト構成ファイルをオリジナルと比較し、構成相違ファイルに相違のみをセーブする。

応答ベクトルメモリを応答回路網に構成する代わりに、設計部コンバータはフォールト応答メモリを構成する。刺激／応答のセクションで説明したように、これらフォールト応答メモリは、応答回路網をメモリ中に記憶された良好な値と比較し、相異が検知される場合にはフリップフロップをセットする。オペレーティングカーネルは、フォールトシミュレーションに対して種々作用する。フォールトシミュレーションを作動させるために（ゼロディレイについて示す。ユニット又はリアル遅延も同様である）：

１）設計部構成ファイルを読出し、これを用いて、すべてのリアライザロジック及び相互接続チップを、構成（configulation）のセクションで説明したように構成する。初期設計部メモリデータを、ファイルから読出し、これを設計部メモリに書込む。構成相違ファイルを読出す。

２）刺激バイナリーファイルを読出す。ベクトルアレイの内容を、ホストインタフェースを介して対応する刺激ベクトルメモリ中に記憶する。タイムアレイ“T”及びサイクルカウント“v”を読出す。良好な回路の応答バイナリーファイルを読出す。対応するフォールト応答ベクトルメモリ中のベクトルアレイの内容を記憶する。

３）この故障の構成の相違を用いて、第１の故障によって影響を受けるロジックチップのフォールト構成ファイルを出力する。またこれらを用いて、この故障に対するロジックチップを構成する。

４）ベクトルメモリモジュール中のすべてのベクトルメモリカウンタ及び相違検出フリップフロップをクリアする。設計部リセット発生器を周期化し、実現される設計部を初期化する。

５）‘v’周期のＥＣＬＫ回路網のクロック発生器をイネーブルする。このことによって、刺激ベクトルメモリはこれらの刺激データを送出し、刺激に応じて実現され

る設計部を作動させることができるとともに、フォールト応答ベクトルメモリは、良好回路に対して応答データを比較する。

6）フォールト応答検出フリップフロップをチェックするとともに、この故障に対して相違が生じたかどうかを記録する。

7）オリジナル構成を、故障したロジックチップに戻す。

8）各残りの故障に対して、ステップ3）～7）を繰り返す。

【００９６】3.3　リアライザロジックシミュレータ評価システム

現在のＥＤＡシステムにおける現行の慣用的なシミュレータの多くは、イベントドライブと称するよく知られたシーケンシャルなアルゴリズム、又はコンパイルされたコードシミュレーションのいずれかに従って作動する（“An Introducion to Digital Simulation”,Mentor Graphics Corp.,Beaverton,Oregon,1989）。第１アルゴリズムにおいては、入力設計部における各基本要素を各時間ステップ毎に“評価”する。この場合、基本要素の入力ピンを駆動する回路網は、イベント即ち状態の変化を有している。また第２アルゴリズムにおいては、全ての時間ステップに対して入力設計部の各基本要素を評価する。基本要素の評価とは、基本要素の新しい出力値が新しい入力値に対してどのようなものであるかを決定する動作のことである。このことは、シミュレーションの間何回も生じる。通常、ゲートのような小さな基本要素のみを１オペレーションで評価する。この際、索引テーブル又は他の直接的な技術を使用する。大規模ロジック回路網は、一般的に小さな基本要素及び回路網の組み合わせとしてシミュレートされる。多くの時間のかかる内部評価は、各大規模回路網の評価毎に必要とされる。リアライザシステムの外部にあり、シーケンシャルなシミュレーションアルゴリズムを実行するロジックシミュレータをリアライザロジックシミュレータ評価システムに結合させる。これは、リアライザのハードウェアを用い、アルゴリズムシミュレーション中の１以上の大規模ロジック回路網を評価する。リアライザシステムによって評価されるべき各大規模ロジック回路網を、外部ロジックシミュレータ中に単一基本要素で表現する。この利益の一つはそのスピードにある。その理由は、実現された基本要素がほとんど瞬間的に評価されるからである。リアライザシステムによって評価されるロジック回路網のサイズは、リアライザのロジック容量によってのみ制限され、全入力設計部と同量のロジック容量を包含している。リアライザロジックシミュレータ評価システムは、リアライザハードウェアシステム及びホストコンピュータと相俟って、リアライザ設計部変換システム（既に記載した）と、リアライザロジックシミュレーションイバリュエータ（evaluator）とからなっている（図５６）。これを、シーケンシャルなシミュレーションアルゴリズムを作動させる外部ロジックシミュレータに結合させる。リアライザロジックシミュレーション評価システムによって、評価のためのロジック回路網を準備するために：

1）リアライザシステムによって評価されるべきロジック回路網をＥＤＡシステムの入力設計部として組み立てる。

2）特性を各ロジック回路網の入出力回路網に組込み、シミュレータ及びサンプラによってそれぞれ駆動されるように指示する。

3）通常の方法でリアライザ設計部変換システムを用い、入力設計部を変換し、ロジック回路網のこの集合体に構成及び対応テーブルファイルを出力する。シミュレーションを行うために、以下の方法に従って、外部ロジックシミュレータを作動させ、シミュレータアルゴリズムを実行させると共にリアライザロジックシミュレーション評価装置も作動させる。：

1）外部シミュレータのデータ構造を構成し、リアライザシステムによって評価されるべき各ロジック回路網毎に単一の基本要素を設ける。

2）設計部の対応テーブルファイルを読み出し、基本要素入出力を、リアライザホストインターフェースバスのこれらのアドレスと関連させる。

3）構成のセクションにて述べたように、設計部の構成ファイルを読み出し、これを用いて全てのリアライザロジック及び相互接続チップを構成する。ファイルから初期設計部メモリデータを読み出し、これを設計部メモリに書き込む。設計部リセット発生器を周期化させ、実現されるロジック回路網を初期化する。

4）初期値を用いて全てのシミュレータを初期化する。

5）外部ロジックシミュレータのシミュレーションアルゴリズムを作動させる。シミュレーションアルゴリズムでは、この方法を用いてリアライザに基づく基本要素を評価する：

1）このシミュレーション時間ステップにおける、この基本要素への全ての入力に対する値を、リアライザロジックシミュレーションイバリュエータに伝送すると共に、この値をロードするために、対応するシミュレータに送る。

2）リアライザロジックシミュレーションイバリュエータにこの基本要素の全ての出力サンプラをチェックするよう指示し、いかなる出力に対する変化であってもシミュレーションアルゴリズムに伝送し直す。

6）シミュレーションの前後において、ユーザが試験及び修正を行うために、ホストインターフェースを介して設計部メモリ内容をアクセスするために、外部ロジックシミュレータのユーザインターフェースシステムの機能を与える。

シミュレーションアルゴリズムをソフトウェア中で実行

する場合、これをリアライザホストコンピュータで実行するとともに、ホストインターフェースを用い、シミュレータ、サンプラ及び設計部メモリをアクセスする。シミュレーションアルゴリズムをハードウェアで実行する場合、ホストコンピュータへの通信リンクを用い、シミュレータサンプラ及び設計部メモリをアクセスする。ハードウェアシミュレータシステムの変更には、シミュレータハードウェアとリアライザのユーザ指定によるデバイス（ＵＳＤ）モジュールとの間の直接接続を用いる。この方法は以下の相違点を伴うものの、上記と同様である：

１）入力設計部の基本要素の入出力に関するシミュレータ及びサンプラを指示する代わりに、これらを、ハードウェアシミュレータの評価ユニットに対応するＵＳＤ基本要素に接続する。

２）ハードウェアシミュレータの評価ユニットを、リアライザのＵＳＤMに電気的に接続する。入力イベントが発生すると、新しい値を直接接続によって実現される基本要素に供給すると共に、ホストを介するのではなく直接接続によって出力応答を捕捉する。このため、かなりの高速評価スピードが得られる。

【００９７】3.4　リアライザプロトタイピングシステム

入力設計部を実現する場合、これを直接設計部のプロトタイプとして実現し、作動させることができる。一般的にリアライザシステムのタイミングディレイは、究極的なハードウェアの実現によるタイミングディレイと一致しておらず、このためプロトタイプはフル設計スピードで作動することはできないが、リアライザベースのプロトタイプによって、ほとんど実時間で設計部は実際に動作することができる。実現される設計部を、リアライザクロック発生器、ホストを介して制御されるシミュレータ、実際にユーザが指定するハードウェアデバイス、実現される仮想計器（以下で説明する）によってシミュレートし、及び／又は、内部ロジック及び／又は設計部メモリ内容によって自己シミュレートする。設計部のオペレーションを、ホスト、実際にユーザが指定するハードウェアデバイス、実現される仮想計器を介して、及び／又は、設計部メモリ内容を調べることによって、コントロールされるサンプラを用いてモニタするとともに分析する。設計者は直接、“ベンチトップ（benchtop）”環境と同様に、実時間で設計部と対話する。リアライザプロトタイピングシステムは、リアライザハードウェアシステム及びホストコンピュータとともに、設計部変換システムと、プロトタイピングシステムとを具えている（図５７）。プロトタイピングオペレータは、作動される設計部のリアライザシステムを構成し、リアライザ設計部の対話型刺激及び応答をサポートする。このオペレータはホストコンピュータにおいて実行し、直接又はホストコンピュータにおいてランする制御プログラムを介して、ユーザのコマンドに対して応答する。実現される設計部を作動させるために：

１）構成のセクションにて述べたように、設計部の構成ファイルを読み出し、これを用いて、全てのリアライザロジック及び相互接続チップを構成する。ユーザが供給するファイルから初期設計部メモリデータを読み出し、これを設計部メモリに書き込む。対応するテーブルファイルを読み出すと共に、設計部回路網ネーム間の対応と、スティミュレータ及びサンプラ及びこれらのホストインターフェースバスアドレスとを確立する。

２）設計部リセット発生器を周期化し、実現される設計部を初期化する。

３）連続的に以下の動作を必要に応じて行う：

―ユーザコマンドを処理し、クロック及びリセット発生器を制御する。

―ユーザコマンドを処理し、スティミュレータ出力値を変化させる。この際、対応テーブルを用い、ユーザが与える回路網ネームを、対応するスティミュレータと関連させる。

―ユーザコマンドを処理し、サンプラのデータ入力値を表示する。この際、対応テーブルを用いユーザが与える回路網ネームを、対応するサンプラと関連させる。

―ユーザコマンドを処理し、設計部メモリモジュール中のロケーションを読み出すと共に書き込む。設計部が動作していないことを確認する。この際、設計部メモリをアクセスする前にクロック発生器が停止し、不適切な設計部メモリの動作を回避することをチェックする。設計部が停止されていないかどうかを、ユーザに報告する。

リアライザプロトタイピングシステムを使用するために：

１）入力設計部をホストＥＤＡシステム中に作成する。

２）シミュレータに接続されるべき設計部回路網と、サンプラと、クロック又はリセット発生器とをマークする。

３）設計部基本要素、回路網及び接続を設け、用いるべき任意の仮想計器に対する回路網を設計する（以下参照）。

４）リアライザ設計部変換システムを用いて入力設計部を変換し、設計部の構成ファイルを出力する。

５）リアライザプロトタイピングオペレータを用いて、設計部を作動させる。図５７にて示す特定の例においては、デジタルコンピュータ設計部を、リアライザプロトタイピングシステムを用いて実現する。ユーザは、ホストＥＤＡシステムを用いて、入力設計ファイル中のコンピュータロジック及びメモリの設計部を表現し、ユーザは、リアライザ設計部変換システムを用いて、構成ファイルへと変換する。実際の具体例においては、実際のフロントパネル制御スイッチ及びインジケータに接続されているフロントパネル制御入力及びディスプレイ出力は、入力設計部において指定され、プロトタイプオペレ

ータを介してのユーザ制御の下、スティミュレータ及びサンプラに接続される。コンピュータのクロック入力信号がリアライザクロック発生器によって出力されるように指定する。プロトタイプコンピュータを作動させるために、ユーザはリアライザプロトタイプオペレータをランさせ、コンピュータ設計に応じてリアライザシステムを構成する。実現されるコンピュータ設計部で実行が可能となるように、コンピュータプログラムコードをロードすると共に、その初期データを、プロトタイプオペレータを介し、動作の開始時に設計部メモリへロードする。ユーザがクロック発生器をイネーブルさせると、コンピュータ設計部は、リアライザハードウェアの構成されたロジック及び相互接続チップにおいて実際に動作し、設計部メモリから読み出されるプログラムインストラクションコードを実行するとともに、設計部メモリ中のデータを読み出し且つ書き込む。ユーザは、フロントパネル制御入力端子を作動させ、プロトタイプオペレータの対応スティミュレータ及びサンプラへのアクセスを介して、動作中ディスプレイ出力を読み出す。結果はプログラムの終了時に、プロトタイプオペレータを介してメモリ中からユーザによって読み出される。ユーザはこの結果を解析し、設計部が正確であるかどうか、すなわちユーザの意図に従って正しく動作しているかどうかを判断する。入力設計部中の設計エラーのために設計部が正しく動作していない場合、ユーザは、ホストＥＤＡシステム用いてエラーを修正し、プロトタイピングプロセスを繰り返す。

【００９８】３．４．１　実現される仮想計器

刺激及び／又は分析計器が、プロトタイプデバッギングプロセスにおいて必要とされる場合、ロジックアナライザのような慣用の計器を、ユーザが与えるデバイスモジュールを介して、実現される設計部に直接接続される。実際の計器を接続するために、計器に接続されるべき設計回路網に接続され、入力設計部中の計器ＵＳＤを表示している基本要素を設けるとともに、ＥＳＤ接続を規定しているＵＳＤ使用ファイルを作成する。このとき、計器を直接ＵＳＤＭに接続し、上記のように実現される設計部を変換させ、作動させる。さらに、“仮想計器”を、入力設計ファイル中の設計部に設け、且つ、この設計部を用いて実現される基本要素及び回路網を設けている。例えばロジックアナライザーを、一組のロジック信号をモニタするよく知られた計器とし、これらが一定のトリガ条件を満足する場合、一組の分析された信号を連続的にサンプリング化するとともに、これらの値をメモリ中に記録する。これはその後分析のために読み出される。図５９は、仮想ロジックアナライザの構成を示し、このアナライザは、応答ベクトルメモリと、ロジック基本要素を有する条件検出器と、１個以上のスティミュレータ及びサンプラと、他のロジック基本要素とを具えている。設計部を用いて仮想ロジックアナライザを実現及び使用するための：

１）設計部に加えて、図で示したように、相互接続された入力設計ファイル中のこれら成分に対する基本要素を設ける。特に、応答ベクトルメモリ入力を、分析されるべき設計部回路網に接続し、条件検出器入力端子を、トリガ条件でモニタされるべき設計部回路網に接続し、これによって、検出されるべき条件に従って条件検出器のロジックを指定する。

２）入力設計ファイルを、通常の手続きに従って構成ファイルに変換する。

３）リアライザプロトタイピングシステム中の設計部を構成する。

４）シミュレータを介して“リセット”信号を周期化させ、実現される設計部が動作を開始するのに必要とされる刺激を与える。

５）“トリガされた”サンプラをモニタする。サンプラが“トリガされた”信号が真であることを示している場合、ロジックアナライザは分析された信号データを捕捉する。

６）このデータを、ロジックアナライザの応答ベクトルメモリからホストインターフェースを介して読み出す。これを、一般的なコンピュータデバッガプログラム又はこれと同様のものを用いて、表示するとともに分析する。これは、いかにして仮想刺激又は分析計器を、リアライザシステム中の設計部を用いて実現するかを示す一例である。ロジックアナライザの概念のような、計器自体の概念が新規でないことに注意する。リアライザシステム中の入力設計部を用いて計器を実現することが、新規性の一要素となっている。

【００９９】３．５　リアライザ実行システム

リアライザ実行システムを用い、入力設計ファイル中で指定され、未だ構成されていない又は永久的ハードウェアにおいて、決して構成することを試みることはないハードウェア機能を実行する。このことを行うことによって、幾つかの利点が得られる：永久的ハードウェアを構成する間、ソフトウェア開発又はその他の目的のために、実現される設計部を使用する。このことによって、例えば、ソフトウェア開発に作成中に行うことが可能となり、これをデバッグし、永久的ハードウェアを使用しない場合にソフトウェアを使用できるように準備する。リアライザ実行システムはユニバーサルハードウェアデバイスとしての役割を果たし、必要とされる種々の互換を行うために用いられる。特別な機能が要求される場合（リアライザ設計変換システムによって実現される場合）、ハードウェアシステムの構成ファイル及びその他のファイルは、ホストコンピュータによって記憶装置から呼び出され、リアライザシステムをこの設計に応じて構成し、機能を実行する。例えば、電気的な設計環境では、リアライザ実行システムを用いて、必要とされるロジックシミュレーションハードウェアアクセレータ、ル

ーティングハードウェアアクセレータ、又はハードウェアグラフィックスプロセッサの役割を果たす。デジタル信号を処理する環境において、リアライザ実行システムを用いて、必要とされる実時間スペクトラムアナライザ又は特別な効果を有するシンセサイザの役割を果たす。リアライザ実行システムは以下の点を除き、リアライザプロトタイピングシステムと同様のものである：

1）分析のための計器を用いず、入力設計を正しいものとみなす。スティミュレータ、サンプラ、及び設計メモリアクセスのみを用いて実行する役割を制御し、データを入出力する。

2）特定の実行される機能を指示するコントローラを作成することができ、又、これを用いてリアライザプロトタイピングオペレータを制御し、実行する機能を、機能の使用に適した入力端子／出力端子及び制御インターフェースに与える。

【０１００】３．６　リアライザ生産システム

リアライザ設計変換システムの変形例を用いて、自動的に入力設計部の永久的で再構成不可能な実現例を作成する。この永久的な実現例では、実現される設計部に構成されるのと同じ種類と数のリアライザロジックチップを用いる。リアライザ生産システムでは、そのＥＲＣＧＡネットリスト変換ツールを用い、機能においてＥＲＣＧＡロジックチップと等価な、永久的で再構成することのできないロジックデバイスを構成するとともに、自動的プリント回路ボード（ＰＣＢ）配置及びルーティングツールを駆動する（“Getting Started with Board Station”，“Layout User's Manual”,Mentor Graphics Corp.,Beaverton, Oregon.1989)。この際、ロジックチップ相互接続に関する仕様を用い、これら再構成することのできないロジックデバイスを永久的に相互接続するＰＣＢを製造する。好適例では、ＬＣＡをＥＲＣＧＡロジックチップとして使用している。ＬＣＡを製造することによって、機能的にＬＣＡと等価な再構成することのできないロジックチップを、構成ＰＲＯＭメモリと結合しているＬＣＡチップの形態で提供する（“The Programmable Gate Array Data Book”,Xilinx, Inc., San Jose,1989)。ＬＣＡネットリスト変換ツールによって、ＰＲＯＭをプログラムするのに用いられるバイナリーファイルを作成する。また、ＬＣＡはロジックを具え、これを用いてＬＣＡが電力を供給する際にＬＣＡ自体を構成することができる。この際、ＰＲＯＭがあればこれを用いる。リアライザ生産システムは、前述したのと同一の設計部リーダと、基本要素コンバータと、リアライザ設計部変換システム（ＲＤＣＳ）、相互接続及びネットリスティングシステム及びＲＤＣＳ中に用いられるものの変形であるＥＲＣＧＡネットリスト変換ツールに使用されるパーティショナと、自動ＰＣＢ配置と、ルーティングツールとを具えている（図６０）。リアライザ生産システムは、リアライザハードウェアシステム又はホストコンピュータを具えていない。これは入力設計部ファイル及びＰＣＢ仕様ファイルを読み出す。以下の方法に従って作動する：

1）設計部リーダを用い、入力設計部ファイルを読み出すとともに、設計部データ構造を作成する。

2）基本要素コンバータを用いて、設計部データ構造をロジックチップ基本要素へと変換する。

3）パーティショナを用い、基本要素を特定のロジックチップに割り当てる。

4）相互接続及びネットリスティングシステムを用い、ロジックチップのためのネットリストファイルを作成する。相互接続チップのためのネットリストファイルを供給する代わりに、カット回路網及びこれらのロジックチップＩ／Ｏピン接続のリストを、自動ＰＣＢ配置及びルーティングツールに受け入れられる形態で単一の相互接続ファイルを送出する。

5）ＥＲＣＧＡネットリスト変換ツールを用い、再構成することのできない等価ロジックデバイスの構成に適した形態で、各ロジックチップ毎にバイナリー構成ファイルを供給する。

6）自動ＰＣＢ配置及びルーティングツールを用いて、相互接続ファイル及びＰＣＢ仕様ファイル（このファイルは、ＰＣＢの寸法、コネクタ必要条件などのようなロジック設計とは直接関係していない物理的な情報を具えている）に読み込み、ＰＣＢ製造データファイルを出力する。リアライザ生産システムのユーザは、このようにしてＰＣＢ製造データファイルを用いて、ＰＣＢを製造し、バイナリー構成ファイルを用いて、再構成することのできないロジックデバイスを構成すると共に、デバイス及びＰＣＢを組立て、入力設計部の最終的な実現例を提供する。リアライザ生産システムにおいて、機能的に永久的ハードウェアの実現例のＥＲＣＧＡと等価な再構成することのできないゲートアレイチップを使用することは新規なことではなく、一般的に行われていることである。むしろ、このシステムが任意の大きさのデジタルシステムを作り出すことができるということ（これは、１個のＩＣチップの容量に限られることではない）、このシステムが入力設計ファイル中で包括的な基本要素ロジックの形態で表現されること（特定のコンピュータメーカのロジックライブラリには限定されない）と、更には、自動的に永久的ハードウェアの実現例を提供するということ、これらが新規性の一態様といえる。

【０１０１】３．７　リアライザ計算システム

リアライザハードウェアシステムを、パスカルのような高級コンピュータ言語で書かれている入力プログラムで特定される動きに従って構成することができる。又これを用いて、コンピュータの実行する一般目的のための記憶された記憶されたプログラムに従って計算機能を実行することができる。このことは、高レベル設計合成コンパイラを用いることによって達成され、コンピュータプ

ログラムを、入力設計ファイル中に表現されているデジタルロジックの形態に変換し、その後リアライザハードウェアにおいて、この設計部を実現すると共に作動させる。この方法は根本的に新しい計算手段である。計算の見地から見れば、リアライザハードウェアを高度並列処理データプロセッサとし、そのデータプロセッシング素子を、リアライザロジックチップ、相互接続チップ及び特定目的のエレメント中のロジック機能及び記憶デバイスである。このデータプロセッサは、シーケンシャルな計測を行うことに関する記憶されたプログラム計算方法に従って演算を行うわけではない。このデータプロセッサは、リアライザハードウェアに構成され、入力プログラムで指示される動きに従って作動するデータパスと、機能的ユニットと、有限状態マシン制御構造とに従って作動する。この利点は、計算スピードがシーケンシャルに記憶されたプログラムによる計算で可能な計算スピードよりも速いことである。説明するリアライザ計算システムは、リアライザハードウェアシステム及びホストコンピュータとともに、リアライザ計算コンパイラと、リアライザ設計部変換システムと、リアライザ計算オペレータとを具えている（図６１）。このホストコンピュータが、ただリアライザ計算オペレータをランさせる手段としてのみ用いられ、入力プログラムで指示される計算機能を実行することに関しては用いられないということに注意する。リアライザ計算オペレータをランさせる他の手段を用いることができること勿論である。

【０１０２】３.７.１　リアライザ計算コンパイラ

リアライザ計算コンパイラは、テキストエディタを用いて高級言語で書かれた入力プログラムファイルを、入力設計部ファイルに変換する。これは、設計部合成コンパイラと、ロジック合成コンパイラと、機能的ユニットコンパイラとを具えている。設計部合成コンパイラは、ツールであり、その幾つかの例は最近開発されたものである（“Tutorial on High-Level Synthesis”,McMarland,Parke and Camposano, Proceeding of the 25th Design Automatic Conference, ACM and IEEE,1988)。このコンパイラは、機能的ユニット、データ入出力から成る有限状態マシンコントローラ及びデータパスのシステムとバス相互接続とに関する記述を構成し、標準的な手続コンピュータ言語で特定される動きに従って作動する。実際の設計部合成コンパイラの一例としては、“フラメル（flamel）”がある。その方法については、“Flame l:A High-Level Hardware Compiler”,Howard Trickey, IEEE Transaction on Computer-Aided Design, Vol. CAD-6, No.2, 1987)で詳細に説明されている。文献からの引用を示す：“フラメルへの入力は、バスカルプログラムである。”“ユーザは、バスカルプログラムに、入力プログラムを一般的に実行する場合の実行頻度を与える。その他のユーザ入力は、どの程度のハードウェアが許容されるかを示すナンバである。出力は、バスカル言語と同一の役割を果たすハードウェアの設計である。”“フラメルによって作り出されるモデルとは、データパス及びコントローラから成る同期デジタルマシンである。データパスは、バスによって相互接続される機能的ユニット（ＡＬＵ、加算器、レジスタ、Ｉ／Ｏバッド等）から成っている。コントローラは有限状態マシンである。”“一般的なバスカルプログラムを用いて、ハードウェアに要求される動きを規定する。フラメルは、プログラム中の並列処理を見出し、ユーザ指定によるコスト制約と合致した高速実行の実現を可能としている。”“フラメルの実現例は完成されている。出力は、データパス及びコントローラに関する記述である。一連のテストにおいて、フラメルは、クロックサイクルを同じとすると、同じプログラムを実行するのにMC68000(マイクロコンピュータ）の２２～２００倍の速さでランするプログラムを具体化する。”用いられるリアライザハードウェアシステムの容量に従って、ユーザ又はリアライザ計算システムは、“ユーザ指定のコスト制約がある”入力を、この設計部合成コンパイラに供給する。設計部合成コンパイラの出力は、データパス及びコントローラの記述を具える中間表現ファイルである。機能的なユニットライブラリとは、一組の予め定義された機能的なユニットの表現である。各タイプの機能的なユニットに対する表現は、設計部合成コンパイラによって与えられる。これらの表現は、ロジック及びユーザ指定のデバイス(ＵＳＤ）基本要素と、これらの回路網相互接続とを指定する。これらの表現は、リアライザ入力設計部基本要素の要件と合致している。ＵＳＤ基本要素は、付加的に用いられ、ロジックチップ及び設計部メモリを用いて実現されるものよりも、より高い性能又は、より大きな容量の基本要素を提供することができる。例えば、高速ＶＬＳＩ浮動小数点型乗算器をＵＳＤとして取り付ける場合、機能的なユニットバイナリーは、このＵＳＤ基本要素を特定する浮動少数点型乗算器の機能的ユニットに関する記述を具えている。ロジック合成コンパイラは、データパス及び有限状態マシンコントローラに関する記述を、入力計算ファイル中のロジック基本要素及び相互接続回路網に関する表現に変換する。このロジック合成コンパイラは、有限状態マシン合成ツールを具え、これは、Mentor Graphics Corp.,VLSI Technology Inc.,Synopsis Inc.等（“Logic Synthesis speeds ASIC Design”, A. J. de Geus, IEEE Spectrum,August 1989）から商業的に入手可能であり、又は、文献記載の方法に従って開発される（“The Implementation of a State Machine Compiler”, C. Kingsley,Proceedings of the 24th Design Automation Conference, ACM and IEEE,1987; ”A State Machine Synthesizer”,D. Brown, Proceedings of the 18th DesignAutomation Conference, ACM and IEEE, 1981; ”An Overview of Logic Synthesis Systems”, L.Trevillyan, Proceedings of the 24th Design Autom

ation Conference, ACM and IEEE, 1987)。このコンパイラは、以下の方法に従って作動する：

1）データパス及びコレクタに関する記述を含む中間表現ファイルを、データ構造中に読み込む。

2）機能的ユニットライブラリの記述に従って、各データパスの機能的なユニットの記述を、ロジック及びＵＳＤ基本要素及び回路網に変換する。

3）データパスへの各データ入力及び、データパスからの各データ出力に対する設計部メモリ基本要素を提供する。

4）有限状態マシン合成ツールを用いて、有限状態マシンコントローラの記述を、ロジック基本要素及びこれらの回路網相互接続に変換する。

5）有限状態マシンコントローラへの‘スタート’入力と、有限状態マシンコトローラからの‘ビジー’及び‘ダーン（done）’出力とに対するスティミュレータ及びサンプラの基本要素を提供する。

6）クロック回路網が、リアライザクロック発生器によって駆動されるように指示する。

7）基本要素及び回路網を、入力設計ファイルに送出する。

【０１０３】３.７.２　リアライザ計算オペレータ

リアライザ計算オペレータは、リアライザシステムを構成し、本来的に入力プログラムが指示し、実現される計算機能の実行を可能にする。リアライザ計算オペレータは、設計変換によって作成される構成ファイル及び対応テーブルファイルに読み込み、ユーザ指定の計算機能への入力データに関するファイルを読み出すとともに、計算機能からの出力データに関するファイルに書き込む。実現される計算機能を作動させるために：

1）設計部の構成ファイルを読み出し、これを用いて、構成のセクションにて述べたように、すべてのリアライザロジック及び相互接続チップを構成する。

2）入力データファイルを読み出し、そのデータを入力データ設計部メモリに書き込む。出力データ設計部メモリをクリアする。

3）対応テーブルファイルを読み出し、コントロール入力と出力との間、スティミュレータとサンプラとの間、及びホストインターフェースバスアドレス間の対応を決定する。

4）クロック発生器をイネーブルし、スティミュレータを介して‘スタート’制御入力を主張し、動作を開始させる。

5）‘ダーン’制御出力をモニタし、これが真となる時に、データを出力設計部メモリから読み出し、これを出力データファイルに書き込む。リアライザ計算システムを用いるために：

1）テキストエディタ又は他の手段を用いて、入力プログラム及び入力データファイルを準備成する。

2）リアライザ計算コンパイラを用い、入力設計ファイルを発生させる。

3）他で既に述べたように、通常の方法で作動するリアライザ設計変換システムを用い、構成及び対応テーブルファイルを発生させる。

4）リアライザ計算オペレータを用い、実際に計算機能を実行する。

5）実現された計算機能によって計算されるデータを、出力データファイルから読み出す。

【０１０４】４　好適例

この明細書を介して説明される好適例は、以下の特徴を有している：

【０１０５】４.１　ハードウェア

部分的クロスバー相互接続を、３レベルで階層的に、全ハードウェアシステムに用いる。図６２—図６４は、階層的に相互接続されたロジックボード、ボックス及びラックの一般的アーキテクチャを示している。図６５ａ—ｂは、ボード、ボックス及びラックに関する物理的構造を示している。

ロジックボード（図６２）：各ロジックボードは、３２個のＸレベルクロスバーチップによって相互接続された１４個のＬチップから成っている。各Ｌチップは、Ｘレベルクロスバーに接続されたチップ毎に１２８個のＩ／Ｏピンを有しており、３２個のＸチップの各々に４個の接続が成されている。１４個の付加的なＩ／Ｏピンを用いる；その内１１個をＲバスに接続し、１個を２個のクロック信号の各々に接続し、１個を設計部リセット信号に接続する。Xilinx XC3090 ＬＣＡをロジックチップとして用いる。各Ｘチップは、ロジックチップに接続された５６個のＩ／Ｏピンを具えており、１４個のＬチップの各々に４個の接続がなされている。各Ｘチップは、２個のＹチップの各々と、８個の付加的Ｉ／Ｏピン接続を具えている。Xilinx XC2018 ＬＣＡをＸチップとして用いる。各ロジックボードは、Ｘ—Ｙバスに対し５１２個の背面Ｉ／Ｏピンを有している。これは、Ｒバス及び構成バスへの接続も有している。

ボックス（図６３）：各ボックスは、６４個のＹレベルクロスバーチップによって相互接続されている１～８個のボードから成っている。各Ｙチップは、ロジックボックズボードに接続された６個のＩ／Ｏピンを有しており、各ボードのＸチップに８個の接続が成されている。これは、１個のＺチップとの８個の付加的なＩ／Ｏ接続を有している。Xilinx XC2018 ＬＣＡをＹチップとして用いる。６４個のＹチップを、８個のＹチップボードに取り付ける。この各々は、Ｘ—Ｙバスに対して５１２個の背面Ｉ／Ｏピンを有している。８個のＹチップボードと８個のロジックボードとを、ボックスのＸ—Ｙバス背面のワイヤで相互接続する。各Ｙチップボードは、Ｙ—Ｚバスに対して、ケーブルコネクタに６４個のＩ／Ｏピンを有している。各ボックスは、このようなコネクタを８個有している。これらの接続を、各ボックスからの単

一な５１２個のワイヤY—Zバスケーブルに集める。各Yチップボードは、構成バスに対する接続も有している。図６５aは、Y—Zバスケーブルを具え、ホストインターフェース、８個のロジックボード及び８個のYチップボードを有しているX—Yバス背面の物理的な構成を示している。

ラック（図６４）：各ラックは、１～８個のボックスを具え、６４個のZレベルクロスバーチップによって相互接続されている。各Zチップは、ボックスに接続された６４個のＩ／Ｏピンを具え、各ボックスのYチップに８個の接続がなされている。Xilinx XC2018 ＬＣＡをZチップとして用いる。ラックのボックスを、ロジックボードに配置された各ボックスからのX—Zバスケーブルへの接続を用いて、付加的なボックスによって相互接続する。図６５bにおいて、Zレベルボックスの物理的な構成を示す。６４個のZチップを、８個のZチップボードに取り付ける。この各々は、Y—Zバスに対して５１２個のＩ／Ｏピンを有している。８個のZチップボードと、８個のY—Zバスケーブルコネクタとを、Y—Zバス背面のトレースによって相互接続する。メモリのセクションにて述べたように、１６個のＲＡＭチップと１０個のＬＣＡとを各々具えているメモリモジュールを、必要とされる場所であるロジックチップの場所に取り付ける。メモリモジュールは、刺激及び応答のセクションで規定されているように、設計部メモリ、ベクトルメモリ、スティミュレータ及びサンプラに用いられている。ユーザ指定によるハードウェアデバイスモジュールをロジックチップＬＣＡの場所に取り付ける。ある１個のボックスは、ホストコンピュータのＩ／ＯバスインタフェースカードとケーブルⅠ接続しているホストインターフェースボードを具えている。このボックスは、Ｒバスと称するホストインターフェースバスを制御する。すべての制御及びデータ伝送機能のためにこのバスをすべてのロジックチップロケーション及び各ロジックボード、すなわちYチップボード及びZチップボードにおける構成制御ロジックブロックに接続する。Ｒバスは、そのセクションで述べたように、８ビットデータバスと、クロックと、２個のコントロールラインとを具えている。ホストインターフェースボードは、構成バスコントローラと、２個のクロック発生器と、リセットコントローラとを具えている。１６ビットデータバスを有する構成バスは、全ての構成機能に対して、ホストインターフェースを用いて、すべてのロジック及びクロスバーチップを接続する。各ボードの１４個のＬチップを一つの構成グループとし、その３２個のXチップを２つのグループに分割する。８個のZチップボードの各々と同様に、各ボックスの８個のYチップボードを各々１グループとする。

【０１０６】４．２ソフトウェア

設計部変換システムは以下のモジュールから成っており、その各々は、それぞれに関するセクションで記載されている：Quick Sim ロジック基本要素を有する Mentor Graphics設計ファイルを読み出す設計部リーダ。Quick Sim 基本要素をXilinx LCA基本要素に変換する基本要素コンバータ。トライステート及びワイヤードネットドライバは、トライステートのセクションで述べたように、クロスバー加算構成に従って変換される。そのセクションにて述べたように、クラスタ構成技術に基づいているパーティショナ。３つのレベルの部分的クロスバーを相互接続するとともに、システム中の各ロジック及びクロスバーチップに関するＸＮＦフォーマットのネットリストファイルを送出する相互接続及びネットリスティングシステム。XNF2LCA, APR及びMakebitsから成っているXilinx ＬＣＡネットリスト変換ツール。構成ファイルコレクタ。

応　用

Mentor Graphics のログファイルに基づき、且つＲＳＩＭバッチインターフェースツールを用いているリアライザロジックシミュレーションシステム。Mentor Graphics のログファイルに基づき、且つＲＳＩＭバッチシミュレーションインタフェースツールを用いているリアライザフォールトシミュレーションシステム。Mentor Graphics Quick Sim のロジックシミュレータとして作用するリアライザロジックシミュレータ評価システム。ロジックアナライザを具え、実現される仮想計器を有しているリアライザプロトタイピングシステム。

リアライザ実行システム

Mentor Graphics Board Station の自動ＰＣＢ配置及びルーティングツールを用いてのリアライザ生産システム。バスカル言語とフラメル設計部合成コンパイラと、Mentor Graphics Design,Knowledge and Logic Consultant FSM及びロジック合成ツールとを用いたリアライザ計算システム。好適例を引用し、本発明の原理を説明したが、このような原理とは離れて、装置及び細部を種々変更できること明らかである。例えば、Mentor Graphics の電気設計オートメーションの変形例を用い、本発明が有効に動作することを説明したが、他の設計部オートメーションツールを用いて同様に本発明を実施できることがわかる。本発明は、ここに開示されている実施例に限定されるものではなく、要旨を変更しない範囲内で種々の変形や変更が可能である。

【図面の簡単な説明】

【図１】図１は、リアライザハードウエアシステムを示す略ブロック図である。

【図２】図２は、直接相互接続システムを示す略ブロック図である。

【図３】図３は、チャンネル・ルーティング相互接続システムを示す略ブロック図である。

【図４】図４は、クロスバー相互接続システムを示す略ブロック図である。

【図５】図５は、クロスバー回路網相互接続システムを

示す略ブロック図である。

【図６】図６は、部分的なクロスバー相互接続システムの簡単な一例を示す略ブロック図である。

【図７】図７は、部分的なクロスバー相互接続システムを示す略ブロック図である。

【図８】図８は、クロスバーチップ幅の相違を説明するための図である。

【図９】図９は、トライステート回路網を示す略ブロック図である。

【図１０】図１０は、図９のトライステート回路網と等価な、積の和を示す略ブロック図である。

【図１１】図１１は、“フローティングロー”及び“フローティングハイ”の積の和の回路網を示す略ブロック図である。

【図１２】図１２は、相互接続を最小とするように構成されたドライバ及びレシーバを示している略ブロック図である。

【図１３】図１３は、ロジック加算構成を示している略ブロック図である。

【図１４】図１４は、クロスバー加算構成を示している略ブロック図である。

【図１５】図１５は、双方向性のクロスバー加算構成を示している略ブロック図である。

【図１６】図１６は、双方向性のクロスバートライステート構成を示している略ブロック図である。

【図１７】図１７は、部分的なクロスバーからのオフボード接続を示している略ブロック図である。

【図１８】図１８は、Ｙレベルクロスバー相互接続を示している略ブロック図である。

【図１９】図１９は、双方向性バスのシステムレベル相互接続を示す略ブロック図である。

【図２０】図２０は、共通バス相互接続に基づく、８個のボードを示す略ブロック図である。

【図２１】図２１は、２つのバスレベルの階層を示す略ブロック図である。

【図２２】図２２は、最大バス相互接続階層を示している略ブロック図である。

【図２３】図２３は、汎用メモリモジュール構造を示す略ブロック図である。

【図２４】図２４は、メモリアドレスロジックチップを示す略ブロック図である。

【図２５】図２５は、共通Ｉ／Ｏを用いてのメモリデータロジックチップを示す略ブロック図である。

【図２６】図２６は、分離Ｉ／Ｏを用いてのメモリデータロジックチップを示す略ブロック図である。

【図２７】図２７は、一個のデータビットに関する多重ＲＡＭを示す略ブロック図である。

【図２８】図２８は、メモリモジュールの好適例を示す略ブロック図である。

【図２９】図２９は、刺激ベクトルメモリを示す略ブロック図である。

【図３０】図３０は、応答ベクトルメモリを示す略ブロック図である。

【図３１】図３１は、刺激及び応答に対するベクトルメモリを示す略ブロック図である。

【図３２】図３２は、ベクトルメモリアドレスチップの好適例を示す略ブロック図である。

【図３３】図３３は、ベクトルメモリデータチップの好適例を示す略ブロック図である。

【図３４】図３４は、ランダムアクセススティミュレータを示す略ブロック図である。

【図３５】図３５は、エッジ感応タイプのスティミュレータを示す略ブロック図である。

【図３６】図３６は、サンプラの略ブロック図である。

【図３７】図３７は、変更検出サンプラを示す略ブロック図である。

【図３８】図３８は、ユーザ指定のデバイスモジュールアーキテクチャを示す略ブロック図である。

【図３９】図３９は、デバイスを取り付けているＵＳＤＭの好適例を示す略ブロック図である。

【図４０】図４０は、構成グループを示す略ブロック図である。

【図４１】図４１は、ホストインターフェースアーキテクチャを示す略ブロック図である。

【図４２】図４２は、Ｒバス読出し及び読込みサイクルを示す図である。

【図４３】図４３は、リアライザ設計変換システムを示す略ブロック図である。

【図４４】図４４は、本発明に用いられている設計部データ構造を示す図である。

【図４５】図４５は、本発明に用いられている基本要素変換を示す図である。

【図４６】図４６は、基本要素のクラスタへの移動を示す略ブロック図である。

【図４７】図４７は、シンプルな回路網相互接続を示す図である。

【図４８】図４８は、トライステート回路網相互接続を示す図である。

【図４９】図４９は、トライステート回路網相互接続を示す図である。

【図５０】図５０は、リアライザロジックシミュレーションシステムを示す略ブロック図である。

【図５１】図５１は、マルチステートロジックのリアライザシステム構成を示す略図である。

【図５２】図５２は、ディレイ依存機能の例を示す略図である。

【図５３】図５３は、ユニットディレイ構成の例を示す略図である。

【図５４】図５４は、リアルディレイ構成を示す略図である。

【図５５】図５５は、リアライザフォールトシミュレーションシステムを示す略ブロック図である。

【図５６】図５６は、リアライザロジックシミュレータ評価システムを示す略ブロック図である。

【図５７】図５７は、リアライザプロトタイピングシステムを示す略ブロック図である。

【図５８】図５８は、リアライザプロトタイピングシステムのデジタルコンピュータの一例を示す略ブロック図である。

【図５９】図５９は、仮想ロジックアナライザの構成を示す略ブロック図である。

【図６０】図６０は、リアライザ生産システムを示す略ブロック図である。

【図６１】図６１は、リアライザ計算システムを示す略ブロック図である。

【図６２】図６２は、ロジックボード、ボックス及びラックの階層相互接続を具える好適例の一般的なアーキテクチャを示す図である。

【図６３】図６３は、ロジックボード、ボックス及びラックの階層相互接続を具える好適例の一般的なアーキテクチャを示す図である。

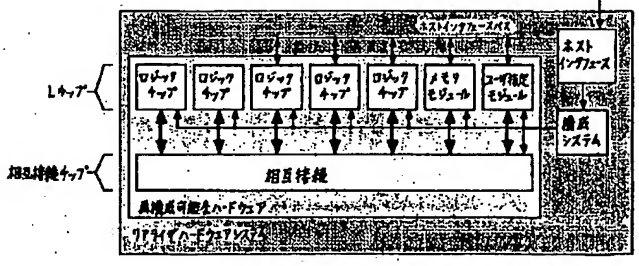
【図６４】図６４は、ロジックボード、ボックス及びラックの階層相互接続を具える好適例の一般的なアーキテクチャを示す図である。

【図６５】図６５は、ロジックボード、ボックス及びZレベルボックスの物理的な構成を示す図である。

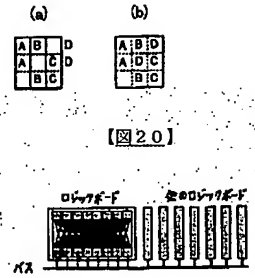
【符号の説明】

１０…アレイ
１２…電気的に再構成可能なゲートアレイ
１４…基本要素コンバータ
１６…ホストＥＤＡシステム
１８…リアライザ設計変換システム
２０…テストベクトル
２２…ホストインタフェース
２４…刺激及び応答ベクトルメモリ
２６ａ…設計合成ツール
２６ｂ…ロジック合成ツール
２８…入力プログラム
３０…ホストインタフェース
３２…メモリモジュール
３４…クロスバーチップ
３６…多次元アレイ

【図１】

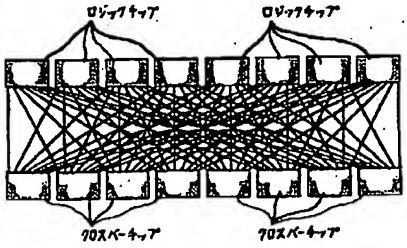


【図８】

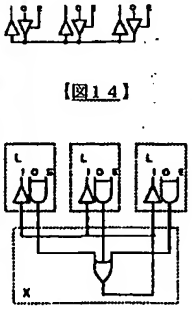


【図２０】

【図６】

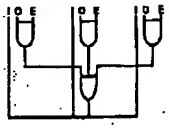


【図９】

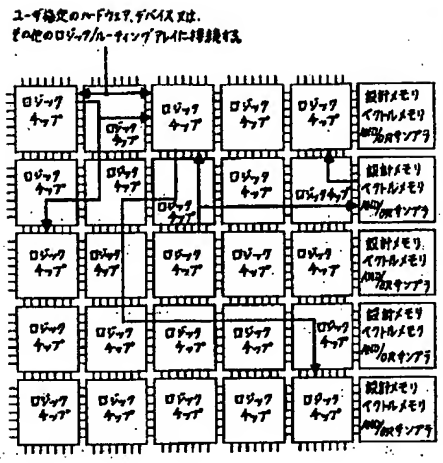


【図１４】

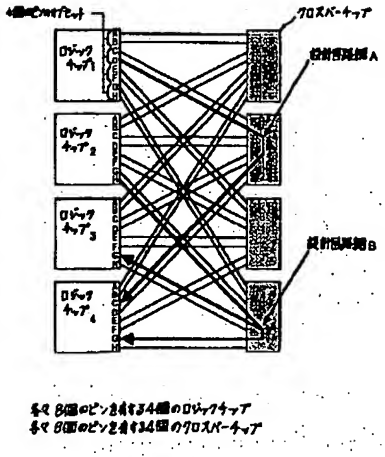
【図１０】



【図２】



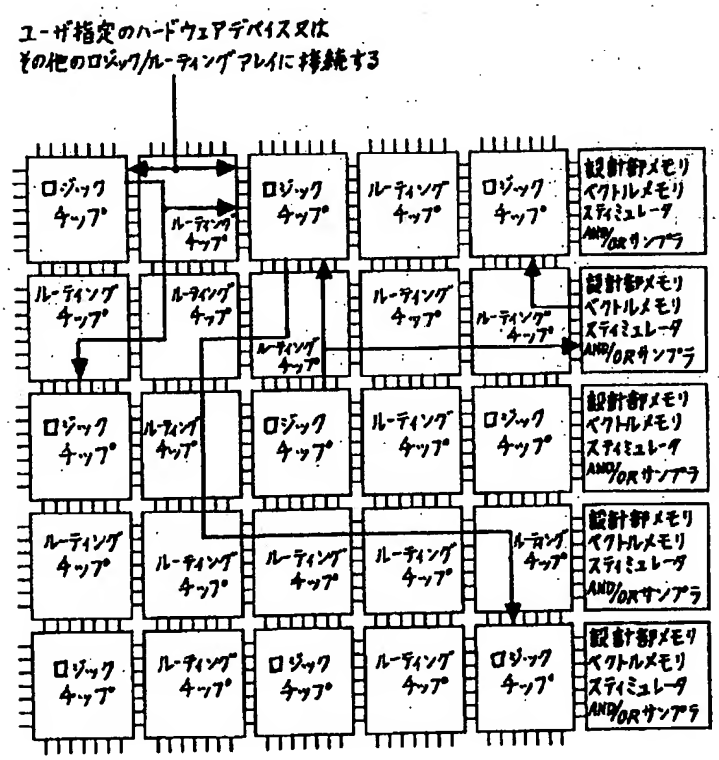
【図７】

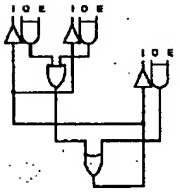


各々 8個のピンを有する4個のロジック チップ
各々 8個のピンを有する4個のクロスバー チップ

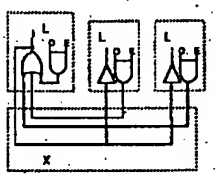
【図１２】

【図３】

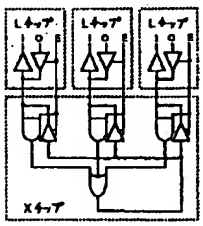




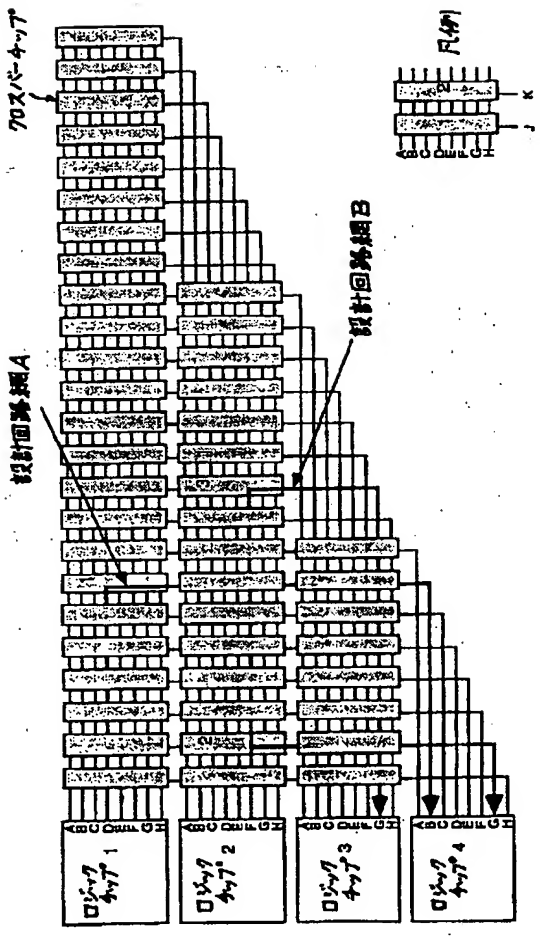
【図１３】



【図１５】



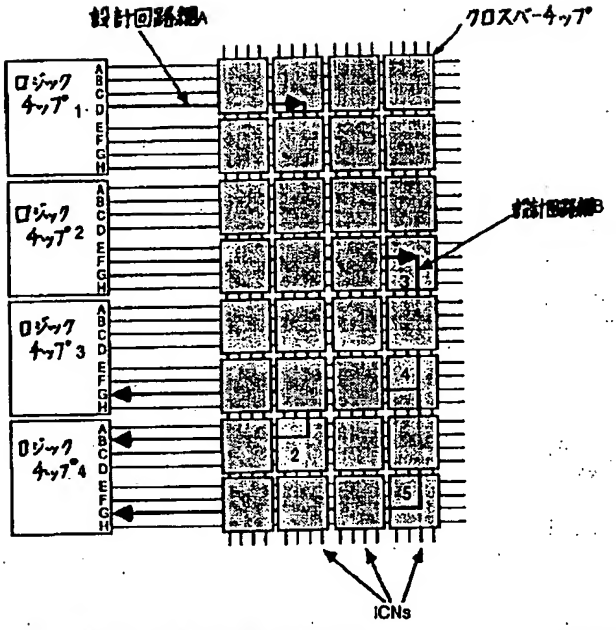
【図4】



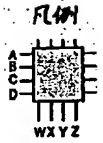
各々8個のピンを有する4個のロジックカップ
各々9個のピンを有する36個のクロスバーカップ

クロスバーカップは、ロジックカップピンJを任意のロジックカップピンA〜Hと接続する。クロスバーカップ2次、ロジックカップピンKを任意のロジックカップピンA〜Hと接続する。

【図５】

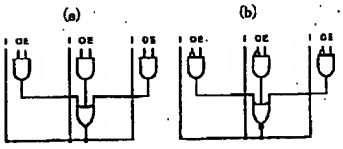


各々 8個のピンを有する4個のロジックチップ
各々 9個のピンを有する48個のクロスバーチップ

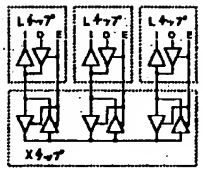


クロスバーチップ1は任意のロジックピンA~Dを、
任意のICN W~Zと接続する。

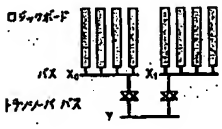
【図１１】



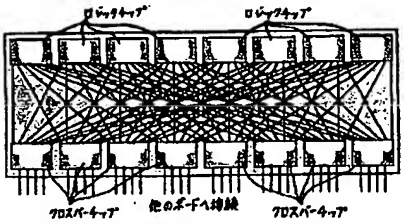
【図１６】



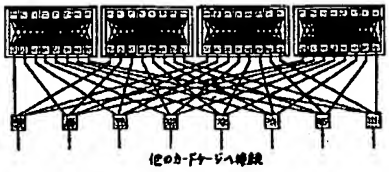
【図２１】



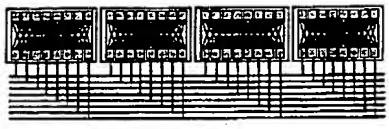
【図１７】



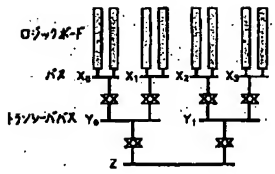
【図１８】



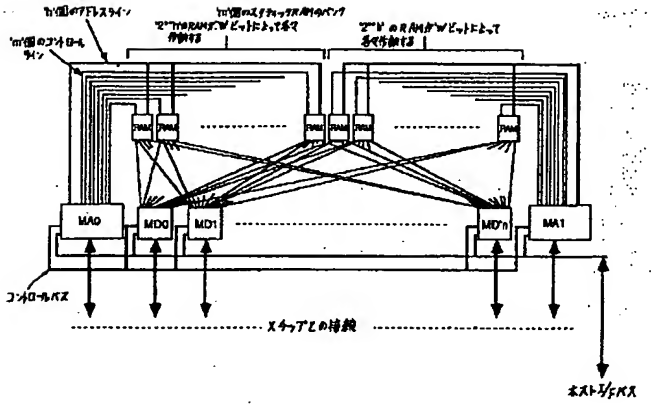
【図１９】



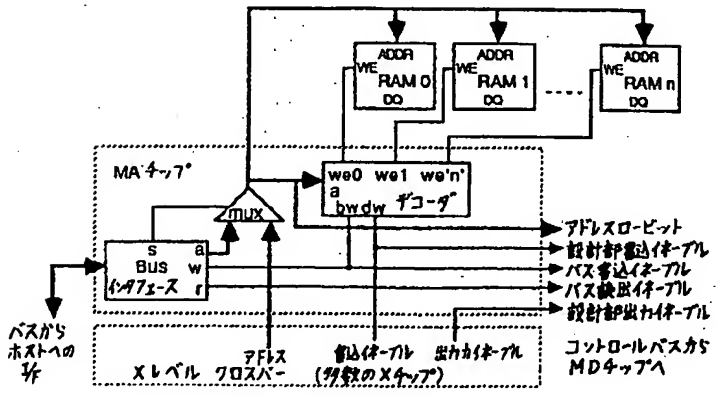
【図２２】



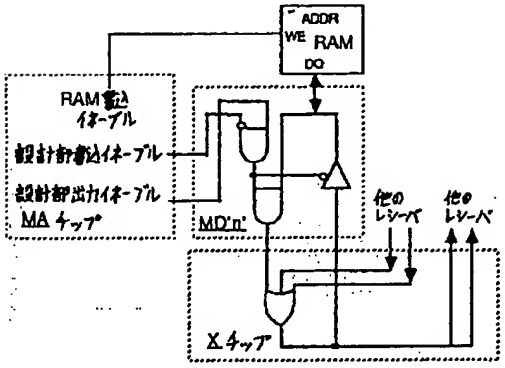
【図２３】



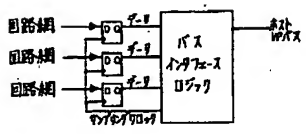
【図２４】



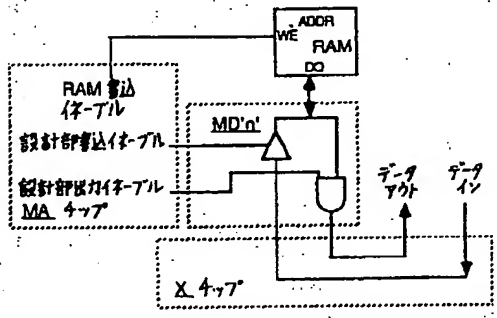
【図２５】



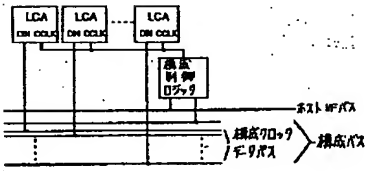
【図３６】



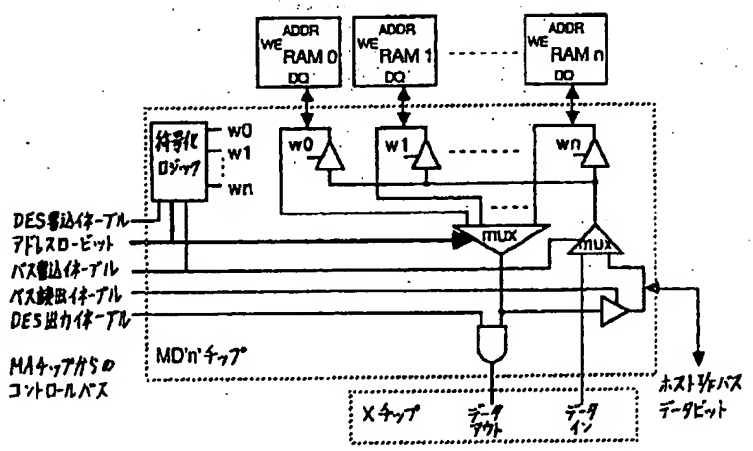
【図２６】



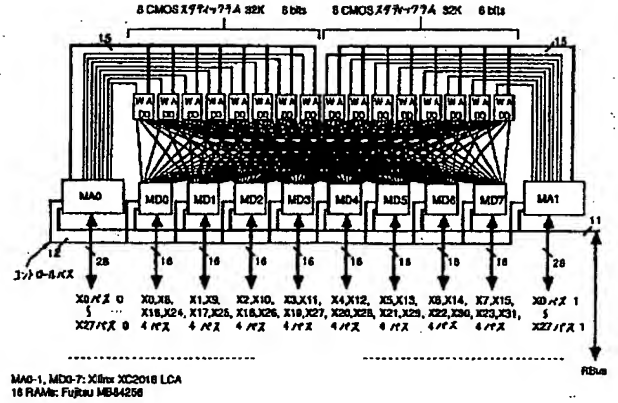
【図４０】



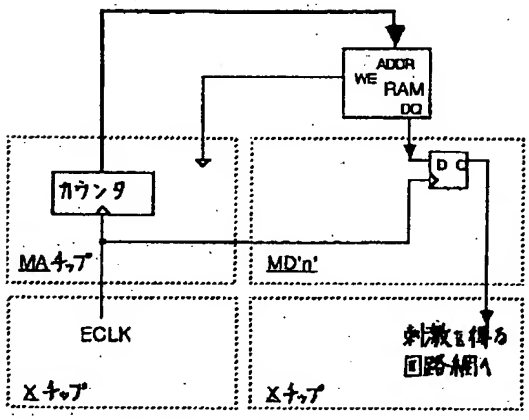
【図２７】



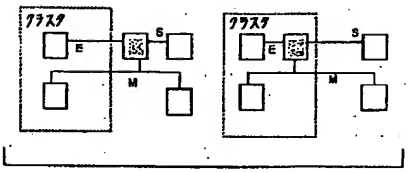
【図２８】



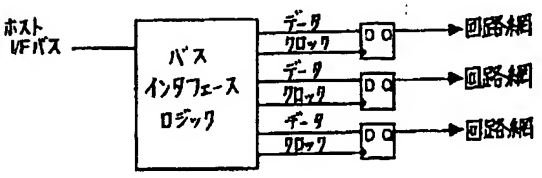
【図２９】



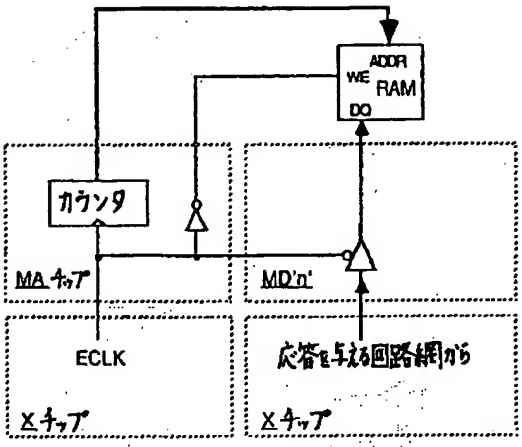
【図４６】



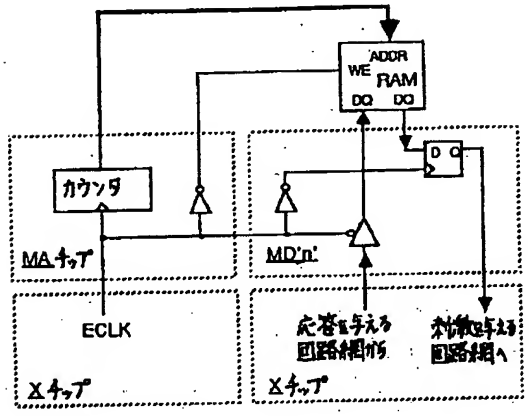
【図３４】



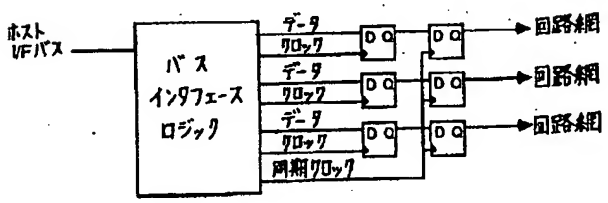
【図３０】



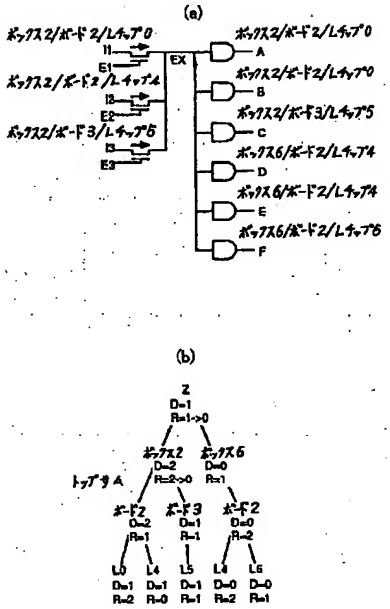
【図３１】



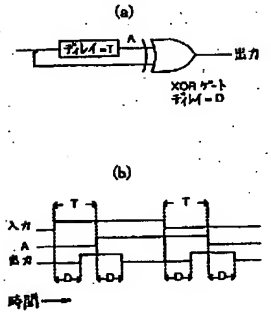
【図３５】



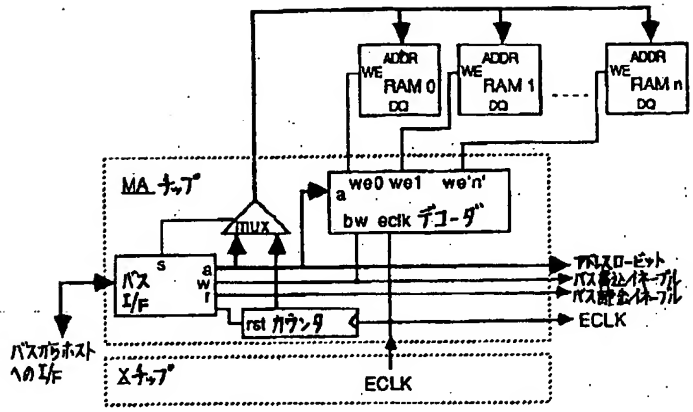
【図４８】



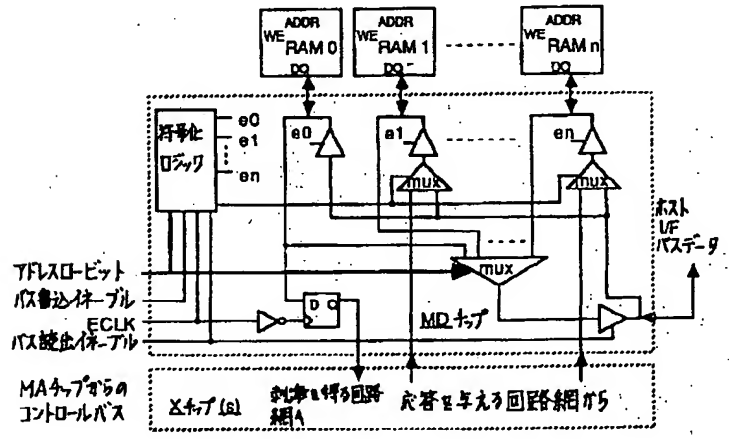
【図５２】



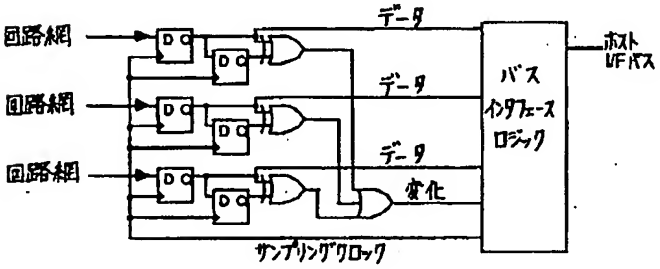
【図３２】


ADDR
WE
RAM 0
DQ
ADDR
WE
RAM 1
DQ
ADDR
WE
RAM n
DQ
MA チップ
we0 we1 we'n'
bw eclk デコーダ
mux
バス I/F
rst カウンタ
ECLK
X チップ
ECLK

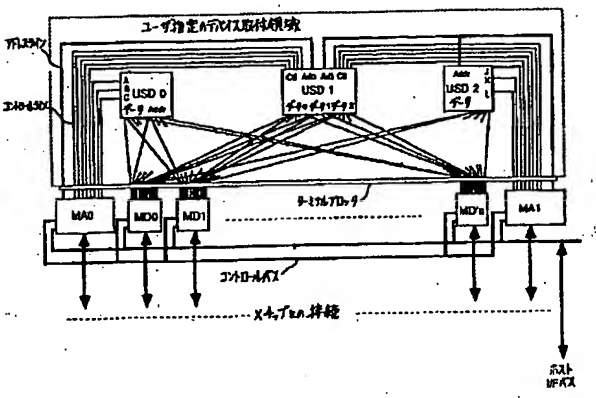
【図３３】


ADDR
WE
RAM 0
DQ
ADDR
WE
RAM 1
DQ
ADDR
WE
RAM n
DQ
待事化ロジック
e0
e1
en
mux
MD チップ
ECLK
X チップ(s)

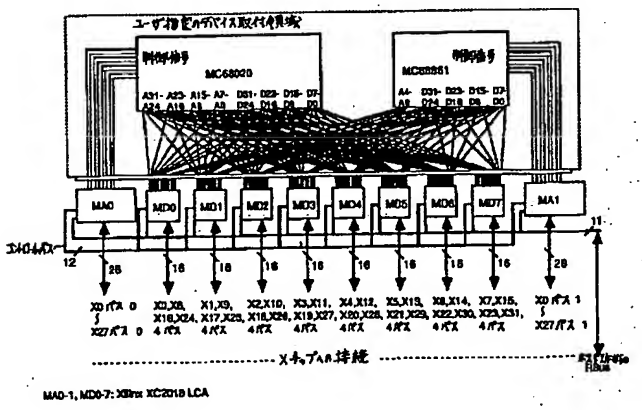
【図３７】


回路網
回路網
回路網
データ
データ
データ
変化
バス
インタフェース
ロジック
サンプリングクロック

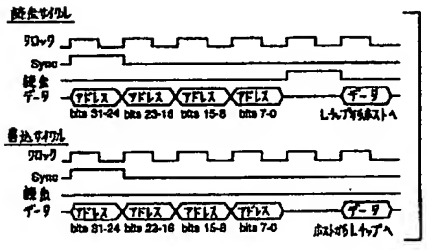
【図３８】


USD 0
USD 1
USD 2
MA0
MD0
MD1
MD's
MA1
ターミナルブロック
コントロールバス
ホスト
MFバス

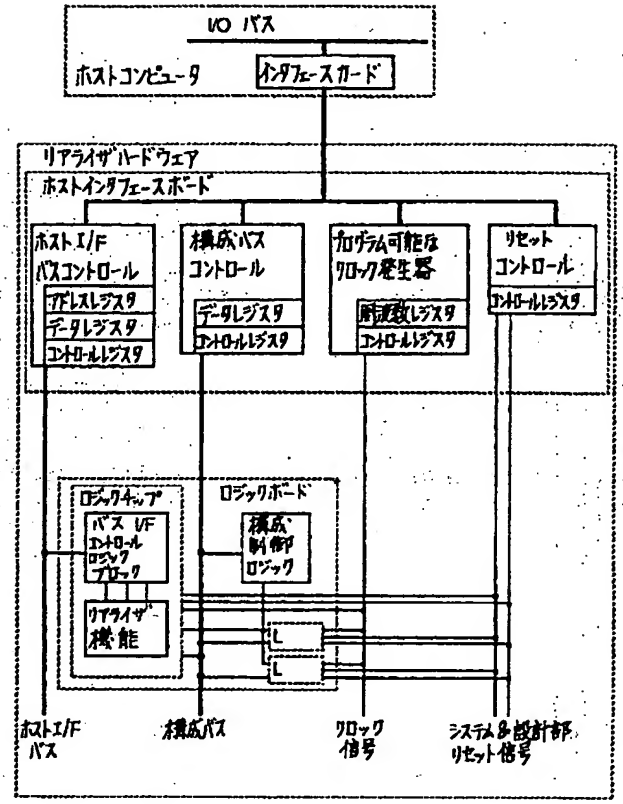
【図３９】


MC68020
MA0
MD0
MD1
MD2
MD3
MD4
MD5
MD6
MD7
MA1
コントロールバス
MA0-1, MD0-7: Xilinx XC2018 LCA

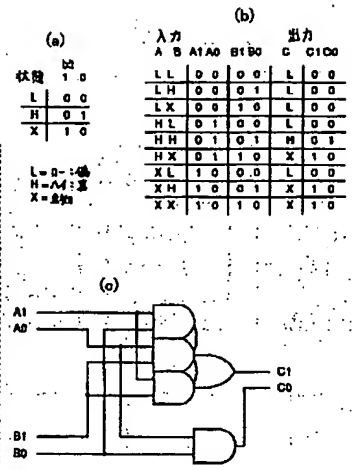
【図４２】


クロック
Sync
アドレス
データ
bits 31-24
bits 23-16
bits 15-8
bits 7-0

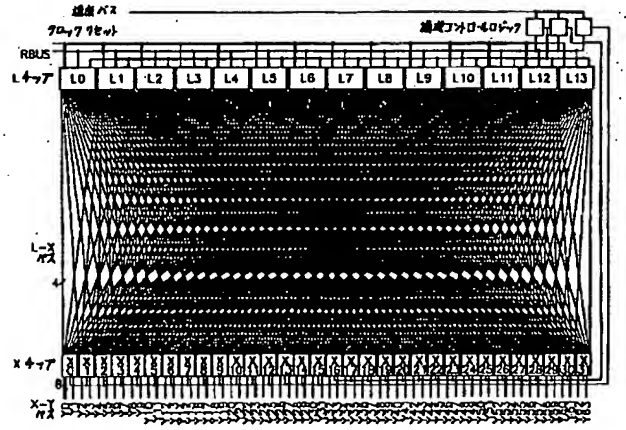
【図４１】


I/O バス
ホストコンピュータ
インタフェースカード
リアライザ ハードウェア
ホストインタフェースボード
ホストI/F
バスコントロール
アドレスレジスタ
データレジスタ
コントロールレジスタ
構成バス
コントロール
データレジスタ
コントロールレジスタ
プログラム可能な
クロック発生器
周波数レジスタ
コントロールレジスタ
リセット
コントロール
コントロールレジスタ
ロジックチップ
バス I/F
コントロール
ロジック
ブロック
リアライザ
機能
ロジックボード
構成
制御
ロジック
L
L
ホストI/F
バス
構成バス
クロック
信号
システム&設計部
リセット信号

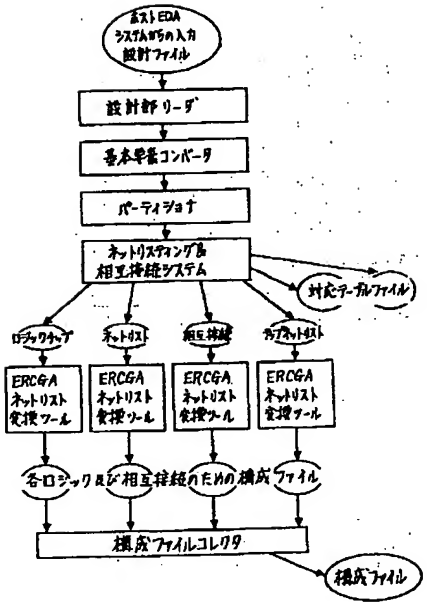
【図５１】


(a)
状態
bit
1
0
L
0
0
H
0
1
X
1
0
L=ロー：低
H=ハイ：高
X=任意
(b)
入力
出力
A
B
A1
A0
B1
B0
C
C1
C0
L L 0 0 0 0 L 0 0
L H 0 0 0 1 L 0 0
L X 0 0 1 0 L 0 0
H L 0 1 0 0 L 0 0
H H 0 1 0 1 H 0 1
H X 0 1 1 0 X 1 0
X L 1 0 0 0 L 0 0
X H 1 0 0 1 X 1 0
X X 1 0 1 0 X 1 0
(c)
A1
A0
B1
B0
C1
C0

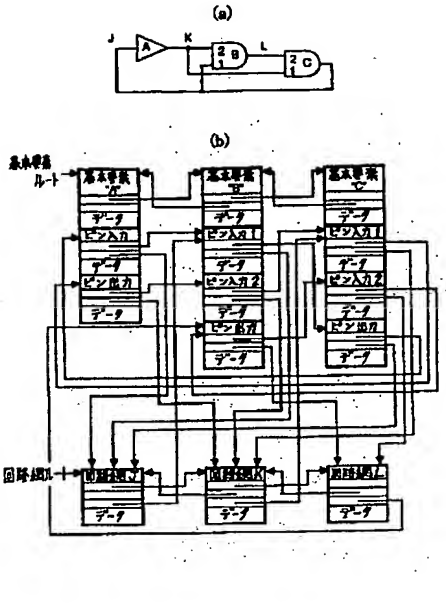
【図６２】


構成バス
クロック リセット
構成コントロールロジック
RBUS
Lチップ
L0
L1
L2
L3
L4
L5
L6
L7
L8
L9
L10
L11
L12
L13
L-Xバス
Xチップ
X-Yバス

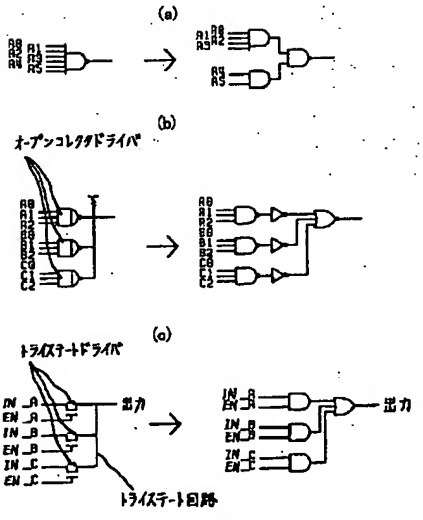
【図４３】



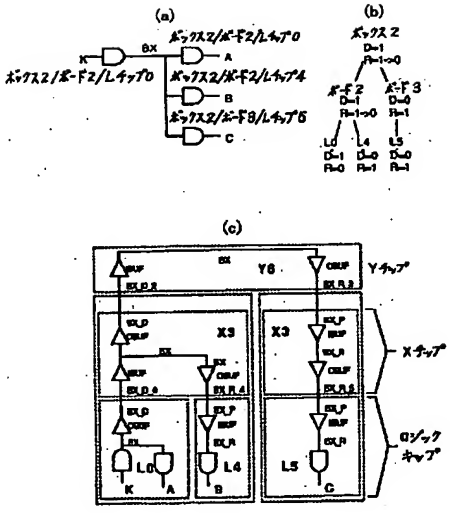
【図４４】



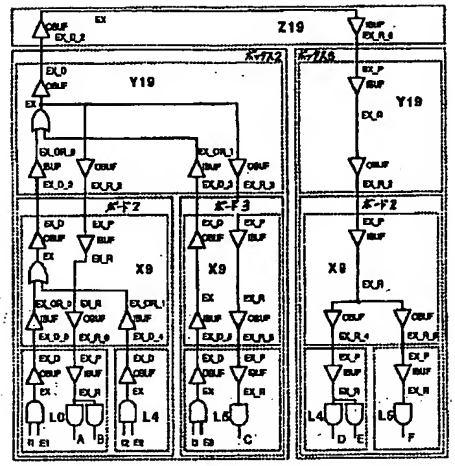
【図４５】



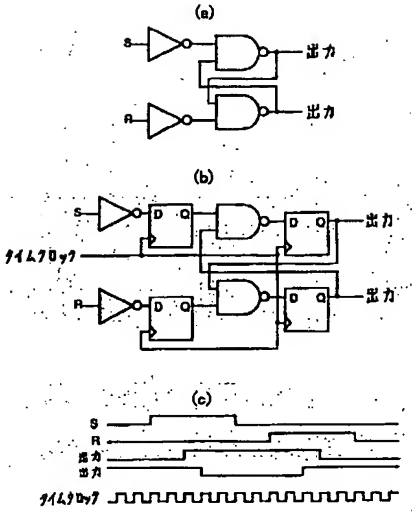
【図４７】



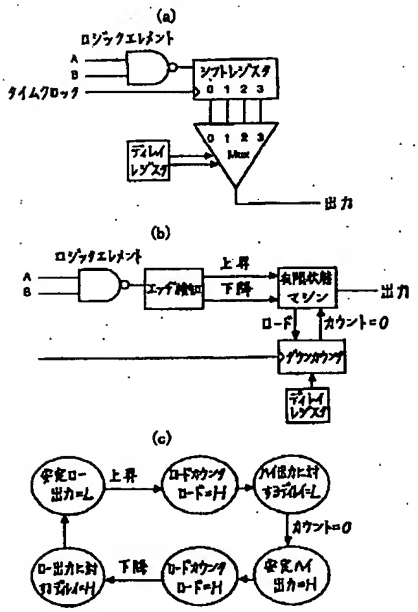
【図４９】


Z19
Y19
X9
L0
L4
L5
A
B
C
D
E
F

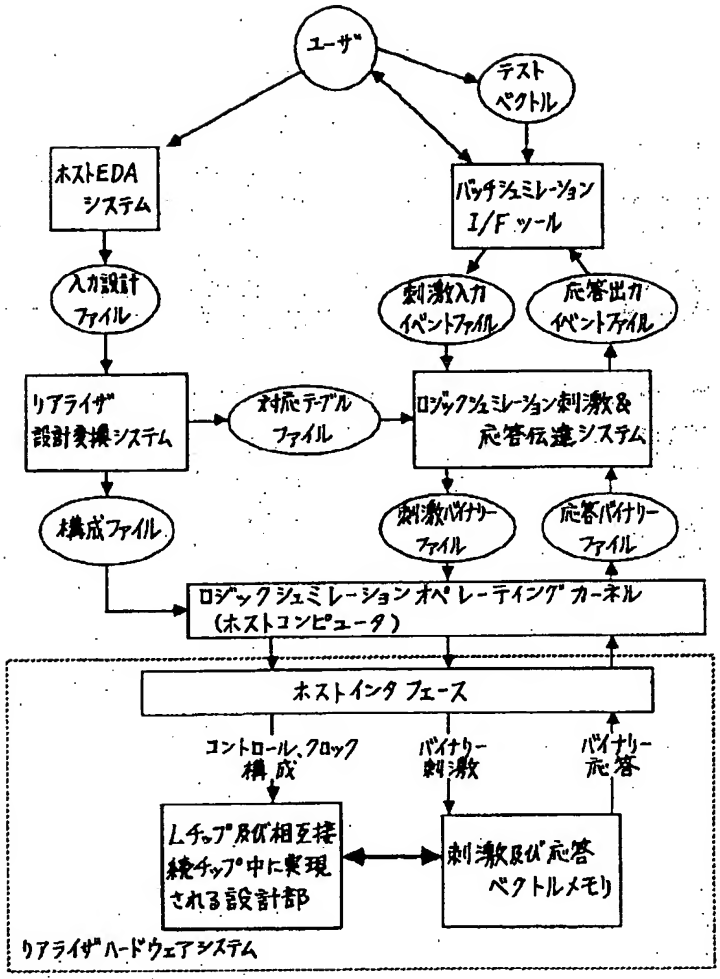
【図５３】


(a)
S
R
出力
出力
(b)
S
R
タイムクロック
D
Q
出力
出力
(c)
S
R
出力
出力
タイムクロック

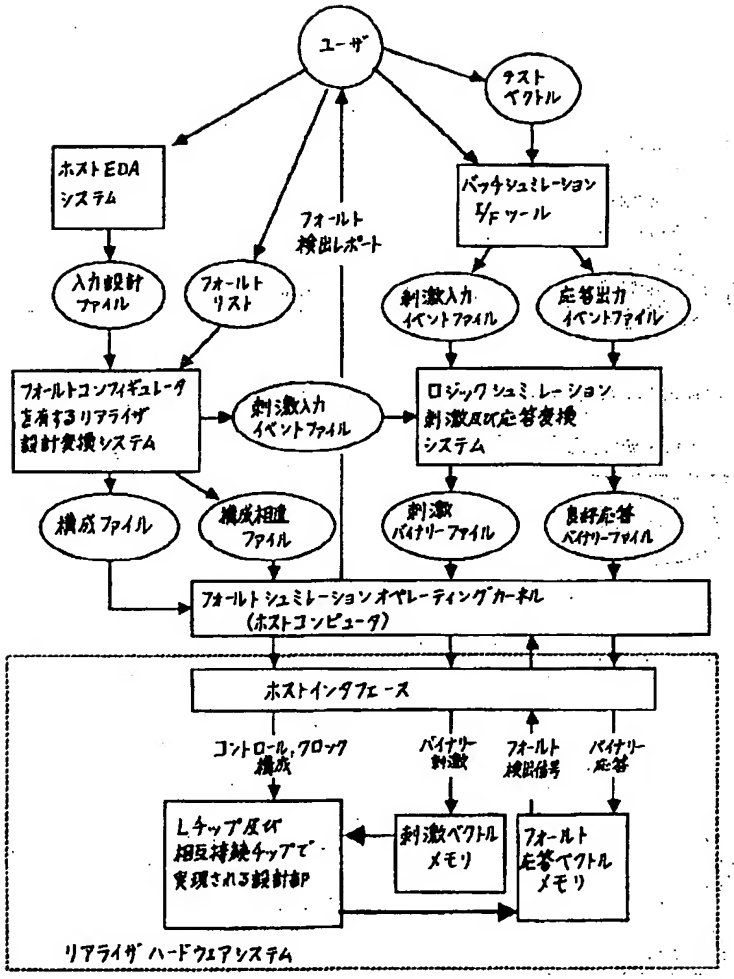
【図５４】


(a)
ロジックエレメント
A
B
タイムクロック
シフトレジスタ
0 1 2 3
0 1 2 3
Mux
ディレイレジスタ
出力
(b)
ロジックエレメント
A
B
上昇
下降
有限状態マシン
出力
ロード
カウント=0
ダウンカウンタ
ディレイレジスタ
(c)
安定ロー出力=L
上昇
ロードカウンタ ロード=H
ハイ出力に対するディレイ=L
カウント=0
ロー出力に対するディレイ=H
下降
ロードカウンタ ロード=H
安定ハイ出力=H

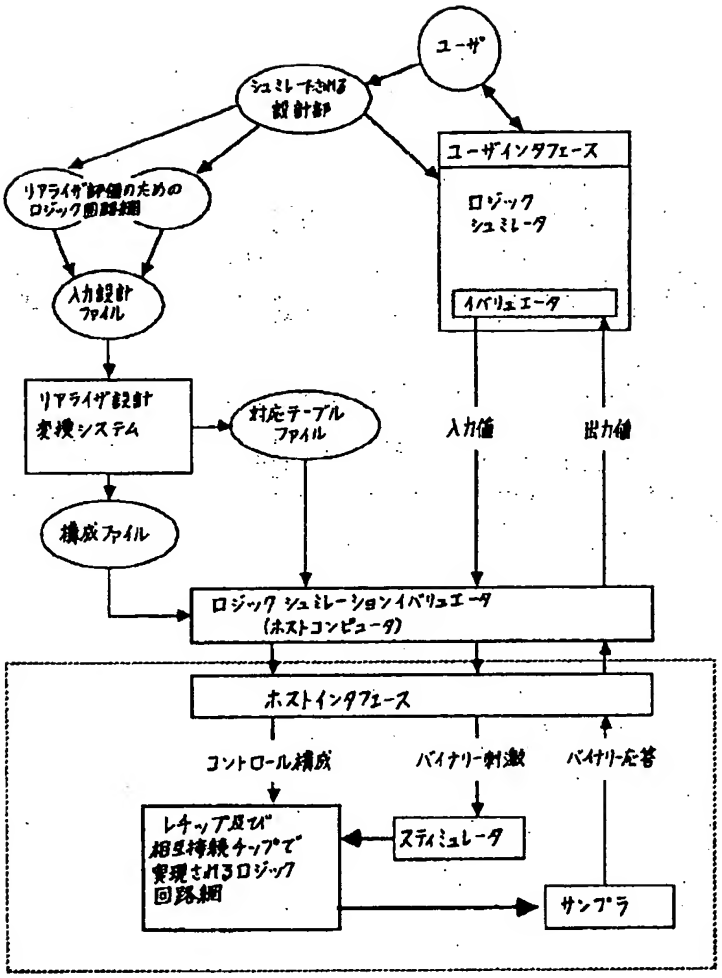
【図50】


ユーザ
テスト
ベクトル
ホストEDA
システム
バッチシュミレーション
I/F ツール
入力設計
ファイル
刺激入力
イベントファイル
応答出力
イベントファイル
リアライザ
設計変換システム
対応テーブル
ファイル
ロジックシュミレーション刺激&
応答伝達システム
構成ファイル
刺激バイナリー
ファイル
応答バイナリー
ファイル
ロジックシュミレーション オペレーティング カーネル
(ホストコンピュータ)
ホストインタフェース
コントロール、クロック
構成
バイナリー
刺激
バイナリー
応答
Lチップ及び相互接
続チップ中に実現
される設計部
刺激及び応答
ベクトルメモリ
リアライザ ハードウェア システム

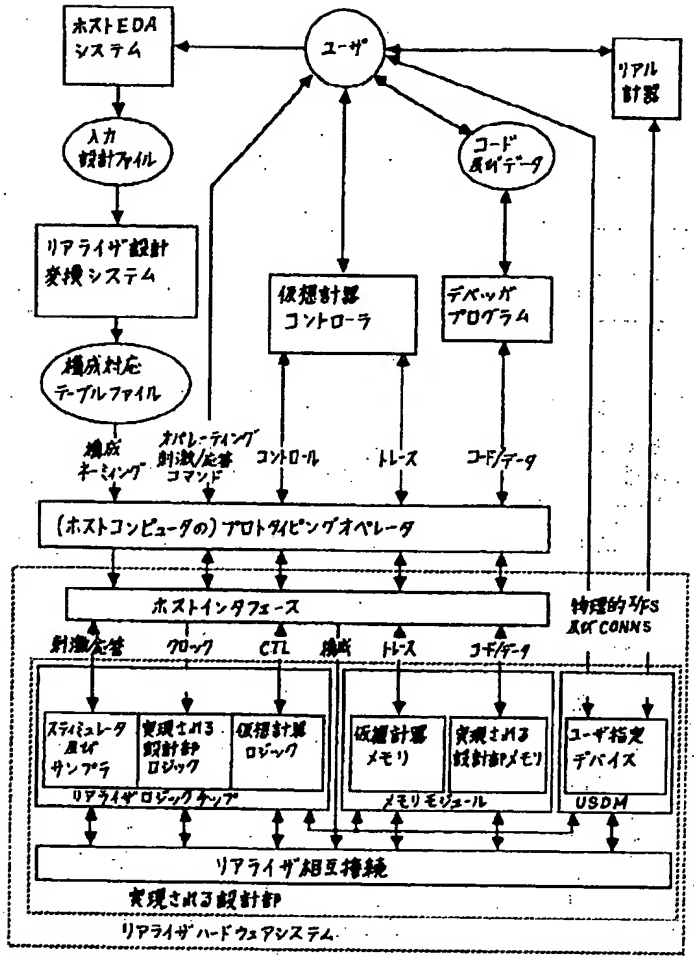
【図５５】


ユーザ
テスト
ベクトル
ホストEDA
システム
フォールト
検出レポート
バッチシュミレーション
I/F ツール
入力設計
ファイル
フォールト
リスト
刺激入力
イベントファイル
応答出力
イベントファイル
フォールトコンフィギュレータ
を有するリアライザ
設計変換システム
刺激入力
イベントファイル
ロジックシュミレーション
刺激及び応答変換
システム
構成ファイル
構成相違
ファイル
刺激
バイナリーファイル
良好応答
バイナリーファイル
フォールトシュミレーションオペレーティングカーネル
(ホストコンピュータ)
ホストインタフェース
コントロール,クロック
構成
バイナリー
刺激
フォールト
検出信号
バイナリー
応答
Lチップ及び
相互接続チップで
実現される設計部P
刺激ベクトル
メモリ
フォールト
応答ベクトル
メモリ
リアライザ ハードウェアシステム

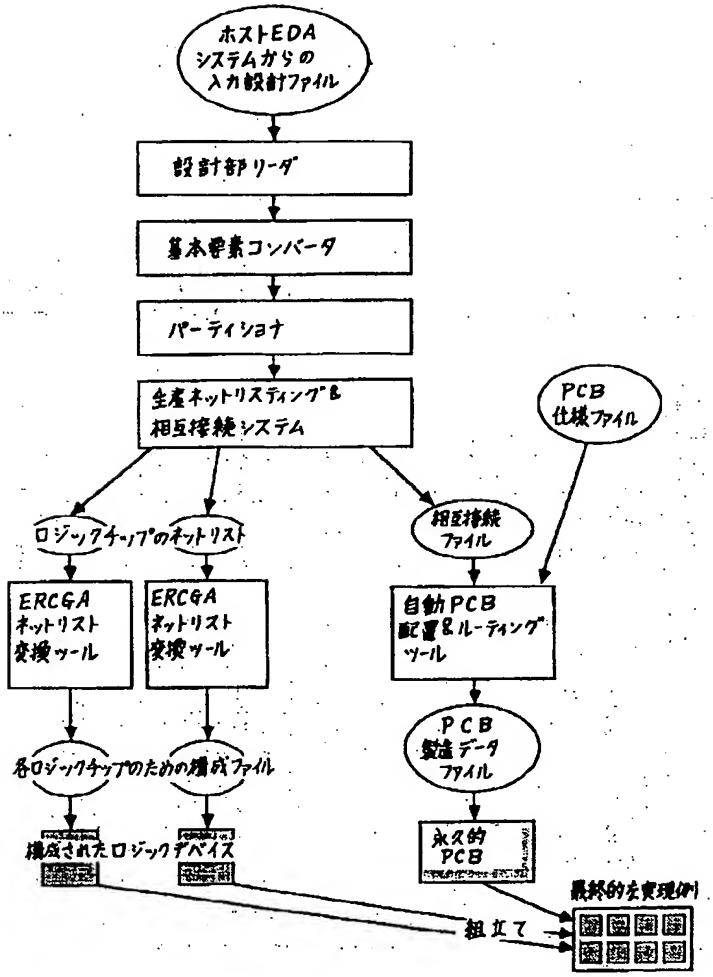
【図５６】


ユーザ
シュミレートされる設計部
ユーザインタフェース
ロジックシュミレータ
イバリュエータ
リアライザ評価のためのロジック回路網
入力設計ファイル
リアライザ設計変換システム
対応テーブルファイル
入力値
出力値
構成ファイル
ロジックシュミレーションイバリュエータ（ホストコンピュータ）
ホストインタフェース
コントロール構成
バイナリー刺激
バイナリー応答
レチップ及び相互接続チップで実現される3ロジック回路網
スティミュレータ
サンプラ

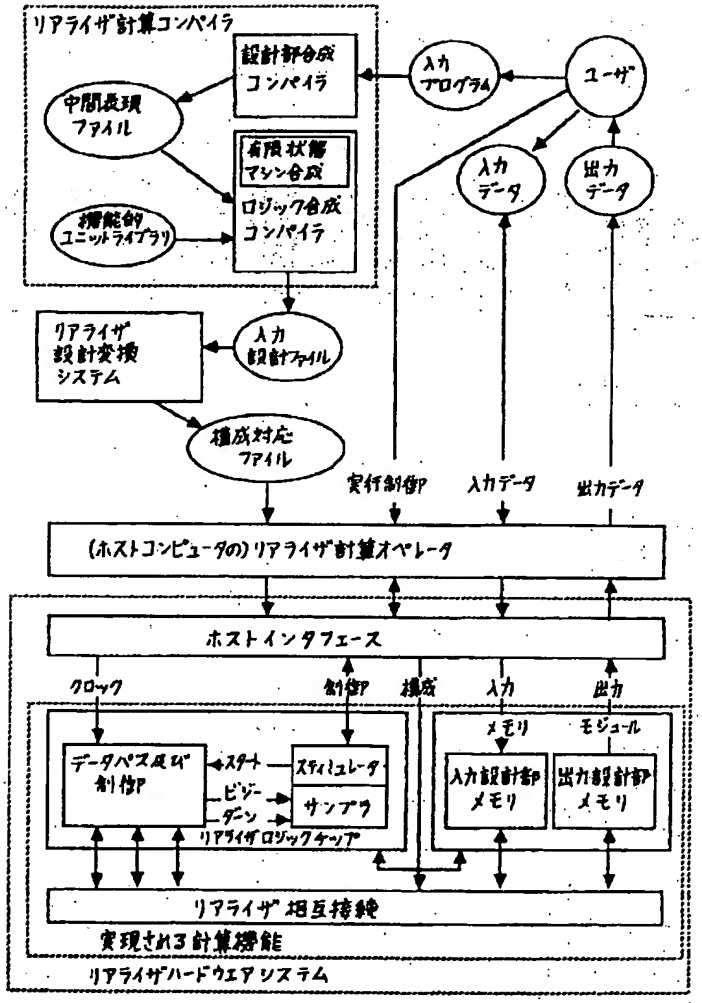
【図５７】


ホストEDA
システム
ユーザ
リアル
計算
入力
設計ファイル
コード
及びデータ
リアライザ設計
変換システム
仮想計算
コントローラ
デバッガ
プログラム
構成対応
テーブルファイル
構成
ネーミング
オペレーティング
制御/応答
コマンド
コントロール
トレース
コード/データ
(ホストコンピュータの) プロトタイピングオペレータ
ホストインタフェース
物理的I/F
及び CONNS
制御/応答
クロック
CTL
構成
トレース
コード/データ
スティミュレータ
及び
サンプラ
実現される
設計部
ロジック
仮想計算
ロジック
リアライザロジックチップ
仮想計算
メモリ
実現される
設計部メモリ
メモリモジュール
ユーザ指定
デバイス
USDM
リアライザ相互接続
実現される設計部
リアライザハードウェアシステム

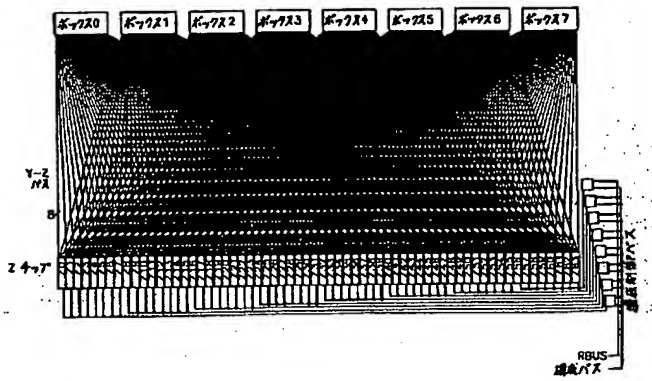
【図６０】


ホストEDA
システムからの
入力設計ファイル
設計部リーダ
基本要素コンバータ
パーティショナ
生産ネットリスティング&
相互接続システム
PCB
仕様ファイル
ロジックチップのネットリスト
相互接続
ファイル
ERCGA
ネットリスト
変換ツール
ERCGA
ネットリスト
変換ツール
自動PCB
配置&ルーティング
ツール
各ロジックチップのための構成ファイル
PCB
製造データ
ファイル
構成されたロジックデバイス
永久的
PCB
最終的な実現体
組立て

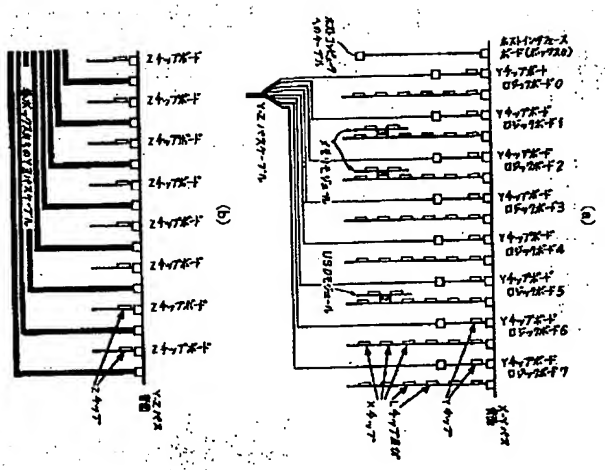
【図６１】


リアライザ計算コンパイラ
設計評価用コンパイラ
中間表現ファイル
有限状態マシン合成
機能的ユニットライブラリ
ロジック合成コンパイラ
リアライザ設計変換システム
入力設計ファイル
構成対応ファイル
入力プログラム
ユーザ
入力データ
出力データ
実行制御P
入力データ
出力データ
(ホストコンピュータの)リアライザ計算オペレータ
ホストインタフェース
クロック
制御P
構成
入力
出力
メモリ
モジュール
データパス及び制御P
スタート
ビジー
ダーン
スティミュレータ
サンプラ
リアライザロジックチップ
入力設計部メモリ
出力設計部メモリ
リアライザ相互接続
実現される計算機能
リアライザハードウエアシステム

【図６４】



【図６５】



【手続補正書】
【提出日】平成１１年６月１１日（１９９９．６．１１）
【手続補正１】
【補正対象書類名】明細書
【補正対象項目名】特許請求の範囲
【補正方法】変更
【補正内容】
【特許請求の範囲】
【請求項１】　回路情報の入力に応じた回路設計に構築することができる電気的に再構成可能なハードウェアシステムにおいて使用するための電気的に再構成可能な論理組立体であって、
複数の再プログラム可能な論理装置と、複数の再プログラム可能な内部接続装置と、固定電気導体のセットとを備え、
前記複数の再プログラム可能な論理装置は、少なくとも組み合わせ可能な論理要素及び記憶要素のグループの中から選択された機能要素を提供するように再プログラム構成可能な内部回路と、前記再プログラム可能な論理装置内に構成された前記機能要素の中から選択された１つに再プログラム接続可能なプログラマブルＩ／Ｏ端子を有し、

前記複数の再プログラム可能な内部接続装置は、Ｉ／Ｏ端子と、前記Ｉ／Ｏ端子の中から選択されたＩ／Ｏ端子間の内部接続を提供するように再プログラム構成可能な内部回路を有し、

前記固定電気導体のセットは、前記再プログラム可能な内部接続装置が前記再プログラム可能な論理装置の前記プログラマブルＩ／Ｏ端子の全てではないが少なくとも１つに接続されるように、前記再プログラム可能な論理装置の前記プログラマブルＩ／Ｏ端子を前記再プログラム可能な内部接続装置の前記Ｉ／Ｏ端子に接続することを特徴とする電気的に再構成可能な論理組立体。

【請求項２】　請求項１に記載の電気的に再構成可能な論理組立体であって、前記再プログラム可能な論理回路は、プログラム可能なゲートアレイにより構成されていることを特徴とする電気的に再構成可能な論理組立体。

【請求項３】　請求項１に記載の電気的に再構成可能な論理組立体であって、前記再プログラム可能な論理装置の前記プログラマブルムＩ／Ｏ端子は、プログラマブルＩ／Ｏ端子のセットに分割され、前記セットの数は、前記複数の再プログラム可能な内部接続装置内の再プログラム可能な内部接続装置の数に等しく、そして前記各プログラムブルＩ／Ｏ端子のセットは、前記複数の再プログラム可能な内部接続装置の１つに接続されていることを特徴とする電気的に再構成可能な論理組立体。

【請求項４】　請求項３に記載の電気的に再構成可能な論理組立体であって、前記再プログラム可能な論理装置と関連付けられた前記プログラマブルＩ／Ｏ端子のセットは、少なくとも２つのプログラマブルＩ／Ｏ端子を有していることを特徴とする電気的に再構成可能な論理組立体。

【請求項５】　回路情報の入力に応じた回路設計に構築することができる電気的に再構成可能なハードウェアシステムにおいて使用するための電気的に再構成可能な論理組立体であって、

複数の再プログラム可能な論理装置と、複数の再プログラム可能な内部接続装置と、固定電気導体のセットと、インターフェース構造とを備え、

前記複数の再プログラム可能な論理装置は、少なくとも組み合わさ可能な論理要素及び記憶要素のグループの中から選択された機能要素を提供するように再プログラム構成可能な内部回路と、前記再プログラム可能な論理装置内に構成された前記機能要素の中から選択された機能要素に接続するようにプログラム可能なプログラマブルＩ／Ｏ端子を有し、

前記複数の再プログラム可能な内部接続装置は、Ｉ／Ｏ端子と、前記Ｉ／Ｏ端子の中から選択されたＩ／Ｏ端子間の内部接続を提供するように再プログラム構成可能な内部回路を有し、

前記固定電気導体のセットは、前記再プログラム可能な内部接続装置が前記再プログラム可能な論理装置の前記プログラマブルＩ／Ｏ端子の全てではないが少なくとも１つに接続されるように、前記再プログラム可能な論理装置の前記プログラマブルＩ／Ｏ端子を前記再プログラム可能な内部接続装置の前記Ｉ／Ｏ端子に接続し、

前記インターフェース構造は、前記再プログラム可能な論理装置内の前記機能要素の指定された機能要素へのあるいは機能要素からの情報搬送信号のための信号通路を提供するように配列されることを特徴とする電気的に再構成可能な論理組立体。

【請求項６】　回路情報の入力に応じた回路設計に構築することができる電気的に再構成可能なハードウェアシステムにおいて使用するための電気的に再構成可能な論理基板であって、

論理基板構造と、複数の論理ゲートアレイと、複数の内部接続ゲートアレイと、固定電気導体のセットとを備え、

前記複数のゲートアレイは、前記論理基板構造上に設けられ、少なくとも組み合わせ可能な論理要素及び記憶要素のグループの中から選択された機能要素を提供するように再プログラム構成可能な内部回路と、前記論理ゲートアレイ内に構成された前記機能要素の中から選択された機能要素に再プログラム接続可能なプログラマブルＩ／Ｏ端子を有し、

前記複数の内部接続ゲートアレイは、前記論理基板構造上に設けられ、Ｉ／Ｏ端子と、前記Ｉ／Ｏ端子の中から選択されたＩ／Ｏ端子間の内部接続を提供するように再プログラム構成可能な内部回路を有し、

前記固定電気導体のセットは、前記内部接続ゲートアレイが前記論理ゲートアレイの前記プログラマブルＩ／Ｏの全てではないが少なくとも１つに接続されるように、前記論理ゲートアレイの前記プログラマブルＩ／Ｏ端子を前記内部接続ゲートアレイのＩ／Ｏ端子に接続することを特徴とする電気的に再構成可能な論理基板。

【請求項７】　設計データによって表わされるデジタル論理回路網設計を形成するための電気的に再構成可能なハードウェアシステムであって、

コンピュータと、複数の再プログラム可能な論理装置と、複数の再プログラム可能な内部接続装置と、固定電気導体のセットとを備え、

前記コンピュータは、前記電気的に再構成可能なハードウェア形成システムへの設計データを受信することができ、形成されるデジタル論理回路網設計を部分に分割する分割コンピュータプログラムと、前記部分間の接続を割り当てる経路コンピュータプログラムと、分割され、接続されたデジタル論理回路網設計を前記電気的に再構成可能なハードウェア形成システム内にプログラムするのに役立つ構成情報を発生する構成コンピュータプログラムを有し、

前記複数の再プログラム可能な論理装置は、前記構成情報を受信可能であり、少なくとも組み合わせ可能な論理

要素及び記憶要素のグループの中から選択された機能要素を提供するように再プログラム構成可能な内部回路と、前記再プログラム可能な論理装置内に構成された前記機能要素の中から選択された機能要素に再プログラム接続可能なプログラマブルＩ／Ｏ端子を有し、
前記複数の再プログラム可能な内部接続装置は、前記構成情報を受信可能であり、Ｉ／Ｏ端子と、前記Ｉ／Ｏ端子の中から選択されたＩ／Ｏ端子間の内部接続を提供するように再プログラム構成可能な内部回路を有し、
前記固定電気導体のセットは、前記再プログラム可能な内部接続が前記再プログラム可能な論理装置の前記プログラマブルＩ／Ｏ端子の全てではないが少なくとも１つに接続されるように、前記再プログラム可能な論理装置の前記プログラマブルＩ／Ｏ端子を前記再プログラム可能な内部接続装置の前記Ｉ／Ｏ端子に接続することを特徴とする電気的に再構成可能なハードウェアシステム。
【請求項８】　請求項７に記載の電気的に再構成可能なハードウェアシステムであって、前記コンピュータは、更に、前記回路情報を含むゲートレベルネットリストを発生するネットリストコンピュータプログラムと、前記ゲートレベルネットリストを分割するように動作する前記分割コンピュータプログラムを有することを特徴とする電気的に再構成可能なハードウェア形成システム。
【請求項９】　回路情報の入力に応じた回路設計に構築することができる電気的に再構成可能なハードウェアシステムにおいて使用するための電気的に再構成可能な論理組立体の配列構造であって、
複数の再プログラム可能な論理装置を含む、少なくとも２つの電気的に再構成可能な論理組立体を有し、
前記再プログラム可能な論理装置は、少なくとも組み合わせ可能な論理要素及び記憶要素のグループの中から選択された機能要素を提供するように再プログラム構成可能な内部回路と、前記再プログラム可能な論理装置内に構成された前記機能要素に再プログラム接続可能なプログラマブルＩ／Ｏ端子を有し、
前記電気的に再構成可能な論理組立体は、第１の複数の再プログラム可能な内部接続装置を有し、
前記第１の複数の再プログラム可能な内部接続装置は、前記再プログラム可能な論理装置の前記プログラマブルＩ／Ｏ端子に接続される第１のグループのＩ／Ｏ端子と、前記第１のグループのＩ／Ｏ端子の中のＩ／Ｏ端子間の内部接続を提供するように再プログラム構成可能な内部回路と、第２のグループのＩ／Ｏ端子を有し、
また、第２の複数の再プログラム可能な内部接続装置を有し、
前記第２の複数の再プログラム可能な内部接続装置は、前記電気的に再構成可能な論理組立体内の前記第１の複数の再プログラム可能な内部接続装置の前記第２のグループのＩ／Ｏ端子に接続され、それによって、前記電気的に再構成可能な論理組立体内の前記複数の再プログラム可能な論理装置の中から選択された論理装置内に構成される選択機能要素が、前記電気的に再構成可能な論理組立体の他の論理組立体内の前記複数の再プログラム可能な論理装置の中から選択された論理装置内に構成された機能要素に再構成可能に接続されることを特徴とする電気的に再構成可能な論理組立体の配列構造。
【請求項１０】　回路情報の入力に応じた回路設計に構築することができる電気的にハードウェアシステム内において使用するための電気的に再構成可能な論理基板の配列構造であって、
少なくとも２つの電気的に再構成可能な論理基板を有し、
前記電気的に再構成可能な論理基板は、複数の論理ゲートアレイを有し、
前記論理ゲートアレイは、少なくとも組み合せ可能な論理要素及び記憶要素のグループの中から選択された機能要素を提供するように再プログラム構成可能な内部回路と、前記論理ゲートアレイ内に構成された前記機能要素に再プログラム接続可能なプログラマブルＩ／Ｏ端子を有し、
また、前記電気的に再構成可能な論理基板は、第１の複数の内部接続ゲートアレイを有し、
前記第１の複数の内部接続ゲートアレイは、前記論理ゲートアレイの前記プログラマブルＩ／Ｏ端子に接続された第１のグループのＩ／Ｏ端子と、前記第１のグループのＩ／Ｏ端子の中のＩ／Ｏ端子間の内部接続を提供するように再プログラム構成可能な内部回路と、第２のグループのＩ／Ｏ端子を有し、
また、第２の複数の内部接続ゲートアレイを有し、
前記第２の複数の内部接続ゲートアレイは、前記電気的に再構成可能な論理基板内の前記第１の複数の内部接続ゲートアレイの前記第２のグループのＩ／Ｏ端子に接続され、それによって、前記電気的に再構成可能な論理基板内の前記複数の論理ゲートアレイの中から選択された論理ゲートアレイ内に構成された選択機能要素が、前記電気的に再構成可能な論理基板の中の他の論理基板内の前記複数の論理ゲートアレイの中から選択された論理ゲートアレイ内に構成された選択機能要素に再構成可能に接続されることを特徴とする電気的に再構成可能な論理基板の配列構造。
【請求項１１】　回路情報の入力に応じた回路設計に構築することができる電気的に再構成可能なハードウェアシステム内において使用するための電気的に再構成可能な論理基板であって、
第１の論理基板構造と、第１の再プログラム可能な論理装置のセットと、第１の再プログラム可能な内部接続装置のセットと、第１の固定電気導体のセットと、第２の論理基板構造と、第２の再プログラム可能な論理装置のセットと、第２の再プログラム可能な内部接続装置のセットと、第２の固定電気導体のセットとを備え、

前記第１の再プログラム可能な論理装置のセットは、前記第１の論理基板構造上に設けられ、少なくとも組み合わせ可能な論理要素及び記憶要素のグループ中から選択された機能要素を提供するように再プログラム構成可能な内部回路と、前記第１の再プログラム可能な論理装置のセット内に構成された前記機能要素の中から選択された機能要素に再プログラム接続可能なプログラマブルＩ／Ｏ端子を有し、

前記第１の再プログラム可能な内部接続装置のセットは、前記第１の論理基板構造上に設けられ、第１及び第２のグループのＩ／Ｏ端子と、前記Ｉ／Ｏ端子の中から選択された端子間の内部接続を提供するように再プログラム構成可能な内部回路を有し、

前記第１の固定電気導体のセットは、前記第１の再プログラム可能な内部接続装置のセット内の前記再プログラム可能な内部接続装置が、前記第１の再プログラム可能な論理装置のセット内の前記再プログラム可能な論理装置の前記プログラマブルＩ／Ｏ端子の全てではないが少なくとも１つに接続されるように、前記第１の再プログラム可能な論理装置のセットの前記プログラマブルＩ／Ｏ端子を前記第１の再プログラム可能な内部接続装置のセットの前記第１のグループのＩ／Ｏ端子に接続し、

前記第２の再プログラム可能な論理装置のセットは、前記第２の論理基板構造上に設けられ、少なくとも組み合わせ可能な論理要素及び記憶要素のグループの中から選択された機能要素を提供するように再プログラム構成可能な内部回路と、前記第２の再プログラム可能な論理装置のセット内に構成された前記機能要素の中から選択された機能要素に再プログラム接続可能なプログラマブルＩ／Ｏ端子を有し、

前記第２の再プログラム可能な内部接続装置のセットは、前記第２の論理基板構造上に設けられ、第１及び第２のグループのＩ／Ｏ端子と、前記Ｉ／Ｏ端子の中から選択されたＩ／Ｏ端子間の内部接続を提供するように再プログラム構成可能な内部回路を有し、

前記第２の固定電気導体のセットは、前記第２の再プログラム可能な内部接続装置のセット内の前記再プログラム可能な内部接続装置が、前記第２の再プログラム可能な論理装置のセット内の前記再プログラム可能な論理装置の前記プログラマブルＩ／Ｏ端子の全てではないが少なくとも１つに接続されるように、前記第２の再プログラム可能な論理装置の前記プログラマブルＩ／Ｏ端子を前記第２の再プログラム可能な内部接続装置の前記第１のグループのＩ／Ｏ端子に接続し、

電気導体のセットは、前記第１の再プログラム可能な内部接続装置のセットの前記第２のグループのＩ／Ｏ端子を前記第２の再プログラム可能な内部接続装置のセットの前記第２のグループのＩ／Ｏ導体に接続することを特徴とする電気的に再構成可能な論理基板。

【請求項１２】　少なくとも２つの再プログラム可能な論理装置と、内部接続手段と、固定電気導体のセットとを備える装置であって、

前記再プログラム可能な論理装置は、組み合わせ可能な論理要素及び記憶要素等の機能要素を提供するように再プログラム構成可能な内部回路と、前記再プログラム可能な論理装置内に構成された選択機能要素にプログラムにより接続可能なプログラマブルＩ／Ｏ端子を有し、

前記内部接続手段は、Ｉ／Ｏ端子と、前記プログラマブルＩ／Ｏ端子の中から選択されたプログラマブルＩ／Ｏ端子間の内部接続を提供するように再プログラム構成可能な内部回路を有し、

前記固定電気導体のセットは、前記再プログラム可能な論理装置の前記プログラマブルＩ／Ｏ端子を前記再プログラム可能な内部接続手段の端子に接続することを特徴とする装置。

【請求項１３】　複数の再プログラム可能な論理装置と、少なくとも１つの内部接続手段と、分割手段と、供給手段とを備える装置であって、

前記内部接続手段は、固定電気導体によって少なくとも２つの再プログラム可能な論理装置に接続されており、

前記分割手段は、デジタル回路設計を表わす入力データを複数のデータ部分に分割し、

前記供給手段は、複数の再プログラム可能な論理装置及び少なくとも１つの内部接続手段内の前記複数のデータ部分を供給することを特徴とする装置。

フロントページの続き

(72)発明者　バチェラー　ジョン　エイ
アメリカ合衆国オレゴン州　97132　ニューバーグ　ボックス　91　ルート　1

PATENT ABSTRACTS OF JAPAN

(11)Publication number : 2000-036737
(43)Date of publication of application : 02.02.2000

(51)Int.CI. H03K 19/173
G06F 17/50
H01L 21/82
H03K 19/177

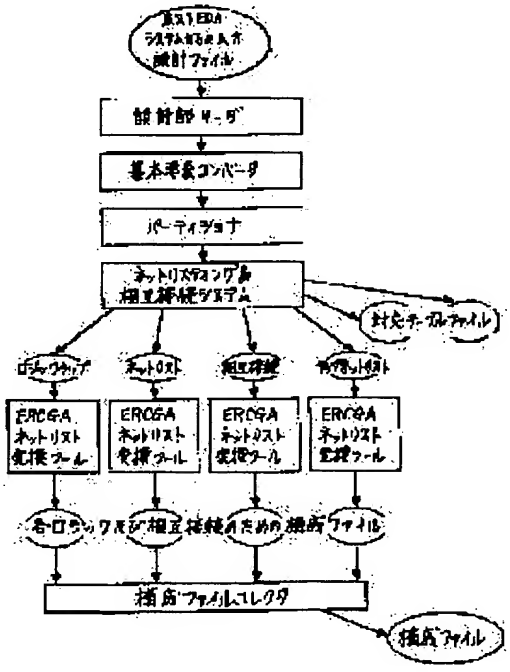
(21)Application number : 11-132028
(22)Date of filing : 04.10.1989

(71)Applicant : QUICKTURN DESIGN SYST INC
(72)Inventor : BUTTS MICHAEL R
BATCHELLER JON A

(30)Priority
Priority number : 88 254463 Priority date : 05.10.1988 Priority country : US

(54) METHOD FOR USING ELECTRICALLY RECONSTITUTABLE GATE ARRAY LOGIC AND DEVICE CONSTITUTED BY THE SAME

(57)Abstract:
PROBLEM TO BE SOLVED: To provide a method for structuring logic constitution by using an electrically reconstitutable gate array.
SOLUTION: Electrically reconstitutable gate array (ERCGA) logic chips are mutually connected through reconstitutable interconnection. The electric representation of a large-scale digital network is so converted as to adopt a hardware configuration, which operates actually and temporarily on interconnected chips. The digital network actualized on the interconnected chips is altered any time through the reconstitution connection. Consequently, a system is adapted to various purposes including simulation, prototyping, implementation and calculation. The reconstitutable interconnection is constituted by an ERCGA chip dedicated to an interconnecting function. The respective interconnected ERCGAs are not connected to all the interconnected chips, but connected to at least one pin.



LEGAL STATUS

[Date of request for examination] 11.06.1999

[Date of sending the examiner's decision of rejection]

[Kind of final disposal of application other than the examiner's decision of rejection or application converted registration]

[Date of final disposal for application]
[Patent number] 3162681
[Date of registration] 23.02.2001
[Number of appeal against examiner's decision of rejection]
[Date of requesting appeal against examiner's decision of rejection]
[Date of extinction of right]

* NOTICES *

CLAIMS

[Claim(s)]
[Claim 1] It is characterized by providing the following and removes. [completely unrelated to said 1st digital logical-circuit network] The 2nd input data expressing the 2nd digital logical-circuit network is supplied. A production process at which said 1st and 2nd digital logical-circuit network takes an actual gestalt of operation in the same ERCGA; said 2nd input data A production process divided into a part for the 1st and part II; a part for part I of the 2nd divided data The amount of [of said 2nd digital logical-circuit network which supplies the 1st ERCGA and is expressed by this] part I A production process which enables it to actually take a gestalt of operation in said 1st ERCGA; a part for part II of said 2nd divided data The amount of [of said 2nd digital logical-circuit network which supplies the 2nd ERCGA and is expressed by this] part II A production process which enables it to actually take a gestalt of operation by said 2nd ERCGA; Said 1st and 2nd ERCGA(s) are interconnected. A method characterized by having production process; which at least one network pinpointed with said 2nd input data constitutes so that it may reach between said 1st and 2nd ERCGA(s) The 1st and 2nd production processes which prepare electrically a gate array (ERCGA) which can be reconfigurated; It has a base element which consists of a Boolean logic gate. A production process which prepares the 1st input data showing the 1st digital logical-circuit network, and a network which interconnects said base element; said 1st input data A production process divided into a part for the 1st and part II; a part for part I of the 1st divided data The amount of [of said 1st logical-circuit network which supplies the 1st ERCGA and is expressed by this] part I A production process which actually takes a gestalt of operation in the 1st ERCGA; a part for part II of said 1st divided data The amount of [of said 1st digital logical-circuit network which supplies the 2nd ERCGA and is expressed by this] part II A production process which actually takes a gestalt of operation in the 2nd ERCGA; Said 1st and 2nd ERCGA(s) are interconnected. A production process which is specified with said 1st input data and at which it is made for a network of a piece to reach between said 1st and 2nd ERCGA(s) at least; a base element which consists of a Boolean logic gate A network which interconnects said base element
[Claim 2] A method according to claim 1 characterized by performing a production process of said segmentation automatically.
[Claim 3] A production process which generates the 1st input data expressing a production process and the 1st digital logical-circuit network of; above which specify the 1st digital logical-circuit network simulated; said 1st input data A production process divided into a part for the 1st and part II; Supply a part for said part I of said 1st divided data to said 1st ERCGA, and the amount of [of said 1st digital logical-circuit network expressed by doing in this way] part I sets it to said 1st ERCGA. A production process which enables it to take a gestalt which actually operates; A part for said part II of said 1st divided data is supplied to said 2nd ERCGA. Thus, a production process at which a part for part II of said 1st logical-circuit network expressed enables it to take a gestalt which actually operates in said 2nd ERCGA; Said 1st and 2nd ERCGA (s) are interconnected. A production process at which it is made for at least one network pinpointed with said 1st input data to reach between said 1st and 2nd ERCGA(s), and a production process which specifies the 1st stimulus of a lot used in the; 1st simulation by

software; software which specifies said stimulus A production process changed into the 1st electrical signal; the 1st electrical signal as said input signal A production process supplied to said 1st and 2nd ERCGA(s) which interconnected; from said 1st and 2nd ERCGA(s) which interconnected A production process which receives the 1st output electrical signal; In 2nd digital logical-circuit network with said 1st digital logical-circuit network another [having production process; which changes said 1st electric-generating-power signal into a gestalt of software] A simulation method [equipped with production process; which repeats the above-mentioned production process] according to claim 1.

[Claim 4] A production process changed into the 1st input data of a lot expressing the 1st digital logical-circuit network which uses a synthetic tool and operates the 1st computer program according to an algorithm expressed by this 1st computer program; said 1st input data A production process divided into a part for said 1st [the] and part II; a part for said part I of said 1st divided data The amount of [of said 1st digital logical-circuit network expressed by supplying said 1st ERCGA and doing in this way] part I A production process which enables it to take ** and an actual gestalt of operation to said 1st ERCGA; A part for said part II of said 1st divided data is supplied to said 2nd ERCGA. Thus, the amount of [of said 1st digital logical-circuit network expressed] part II sets to said 2nd ERCGA. A production process which enables it to take an actual gestalt of operation; Said 1st and 2nd ERCGA(s) are interconnected. A production process which generates the 1st stimulus signal corresponding to input data of a production process at which it is made for a network of a piece to reach between said 1st and 2nd ERCGA(s) at least and the 1st program of; above as which said 1st input data specifies; said 1st stimulus signal A production process supplied to said 1st and 2nd ERCGA(s) which interconnected as an input signal; Interconnected. It is the count method [equipped with production process; which repeats the above-mentioned production process in 2nd digital circuit network with said 1st digital circuit network another / having production process; which receives the 1st output electrical signal corresponding to output data of said 1st program from said 1st and 2nd ERCGA(s)] according to claim 1.

[Claim 5] Said synthetic tool use production process uses :layout section composition tool. Said 1st computer program Consist of a data path and a finite-state machine controller, and a production process and; logic composition tool which are changed into an expression of a system which operates according to an algorithm expressed by said 1st program are used. A method according to claim 1 characterized by having production process; which changes into the 1st input data of a lot an expression of a data path offered by said layout section composition tool, and a finite-state machine controller.

[Claim 6] said ERCGA -- each and two or more pins -- having -- and a production process of said interconnect -- : -- a production process and; which prepare ERCGA of an addition of a piece at least and play a role of interconnect which can be reconfigurated -- a method according to claim 1 characterized by having production process; which connects to a piece at least each of interconnect ERCGA in which said reconstruction is possible although it is not all the pins of said 1st and 2nd ERCGA(s).

[Claim 7] (a) A production process which prepares ERCGA of N individual;

(b) A production process which divides said 1st input data into a portion of N individual;

(c) A production process at which said portion of said digital logical-circuit network expressed by supplying each portion of divided data to corresponding ERCGA, and doing in this way enables it to take an actual gestalt of operation in said ERCGA;

(d) A production process which realizes each of a network which interconnects ERCGA of N individual, and connects each of ERCGA to other ERCGA(s) of a piece at least, and is pinpointed with said input data;

(e) A method according to claim 1 characterized by having further production process; which repeats a production process of (b), (c), and (d) to said 2nd input data.

[Claim 8] a pin of each plurality [ERCGA] -- having -- and a production process of said interconnect -- : -- a production process and; which prepare ERCGA of an addition of a piece at least and play a role as interconnect which can be reconfigurated -- a method according to claim 7 characterized by to have production process; which connects to an individual at least

each: of interconnect ERCGA in which said reconstruction is possible although it is not all the pins of ERCGA of N individual which is said plurality.
[Claim 9] A method given in ******** 8 characterized by having further a production process which connects to a piece at least interconnect ERCGA in which said reconstruction is possible although it is not each ERCGA(s) of all of said N individual.
[Claim 10] rather than it calls it sequential -- a rather single process -- : -- a method according to claim 9 which is further equipped with a production process which performs production process; which identifies the feature of interconnect which corresponds and is needed with a production process and this [;] which divide said input data, divides said input data by such method by this, and is characterized by corresponding and simplifying required interconnect.
[Claim 11] A seed base element is determined and said input data is divided by adding other base elements to this. By this It has further a production process which constitutes a cluster of a base element. Each of the; aforementioned base element many pins -- having --; -- a production process as which a configuration of said cluster estimates an effective function of each base element which is not assigned to a cluster -- having --; -- a method according to claim 10 that said effective function is characterized by bringing the greatest initial profits to a base element which has many pins most.
[Claim 12] Until it reaches a production process which divides said input data and constitutes a cluster of a base element by this by adding other base elements to **** and others, and a limit of; interconnect, while determining a seed base element A method according to claim 10 characterized by having how it has further a production process which removes a base element from a cluster, and a production process of the; aforementioned division adds a base element a to a cluster across a limit of interconnect.
[Claim 13] The production process which examines the interconnect ERCGA from which a network where a production process of said interconnect has reached between :2 piece ERCGA (s) further is set as two or more objects which can determine the root of :network, and which can be reconfigurated; the method according to claim 10 characterized by to have production process; which evaluates the fitness of root decision ERCGA minds each of such interconnect ERCGA partially based on a degree of already used use at least.
[Claim 14] (a) A production process which prepares ERCGA of N individual;
(b) A production process which determines ERCGA which arranges ERCGA of said N individual to a regular multidimensional array, and adjoins relatively by this in circuit gestalt;
(c) A production process which interconnects ERCGA which adjoins directly;
(d) A production process which divides said 1st input data into a portion of N individual;
(e) A production process at which said portion of said digital logical-circuit network expressed by supplying each portion of said divided data to corresponding ERCGA, and doing in this way enables it to take a gestalt which actually operates in said ERCGA;
(f) A production process which interconnects un-adjoining [ERCGA] and realizes a network which interconnects ERCGA of N individual by which necessity is carried out, and is pinpointed by said 1st data by establishing interconnect through ERCGA which intervenes between un-adjoining [ERCGA];
(g) A method according to claim 1 characterized by having further production process; which repeats a production process (d), (e), and (f) to said 2nd input data.
[Claim 15] A method according to claim 14 characterized by having further a production process which determines which ERCGA and pin are used using an automatic routing method. [that intervene although it interconnects un-adjoining / ERCGA]
[Claim 16] A method about a fault simulator according to claim 1 characterized by having further a production process which simulates failure by expressing said digital logical-circuit Aminaka's failure with said input data.
[Claim 17] A method according to claim 1 characterized by having further a production process which operates ERCGA linked to an electrical-design automation system which interconnected.
[Claim 18] A method according to claim 1 further equipped with a production process which combines said ERCGA which interconnected with a memory circuit, and a production process which operates said ERCGA linked to said circuit which interconnected.

[Claim 19] A method according to claim 1 characterized by having further a production process which interconnects a bilateral circuit network by changing said bilateral circuit network into the sum of a product using bidirection interconnect.
[Claim 20] A method according to claim 1 characterized by having further a production process which adds a product in ERCGA.

[Translation done.]

* NOTICES *

1.This document has been translated by computer. So the translation may not reflect the original precisely.
2.**** shows the word which can not be translated.
3.In the drawings, any words are not translated.

DETAILED DESCRIPTION

[Detailed Description of the Invention]
[0001]
[Field of the Invention] This invention equips with Seki, then interconnect of such two or more logic elements both electrically use of the gate array logic element (ERCGA) which can be reconfigurated, and relates also to the method of changing the electric expression of a large digital circuit network into the configuration of the hardware which actually operates temporarily using the logic element for which it interconnected for a simulation, prototyping, activation, and/or an operation.
[0002]
[The background and outline] of invention The expedient top of explanation and this application explain this invention as a rear riser system (Realizer System). The easy explanatory name of the system mentioned later lacks in the dictionary. The rear riser system is equipped with hardware and software, and changes the expression of a large digital logical-circuit network into the configuration of the hardware which actually operates temporarily for a simulation, prototyping, activation, or an operation. (When it has very many logic functions by using the logic device which can use some most widely and which can be constituted, the digital logical-circuit network is made into large **********.) it shall be easier to understand the following explanation by re-evaluating simply the suitable term generally used (chiefly -- coming out -- there is nothing) It is making actual or actual it "realizing" something. Realizing a digital logical-circuit network, the whole layout, or a part is constituting actual actuation, without assembling it eternally. The "input design section" expresses the digital logical-circuit network which should be realized. This input design section is equipped with the base element showing combination logic and storage, and the network expressing connection between the I/O pins of a base element like the measurement device or the user assignment real device. It is arranging the internal logic function and/or interconnect by the specific method as "constituting" a logic chip or an interconnect chip. It says arranging the internal logic function and interconnect according to the input design section as constituting the rear riser system for the input design section. It is changing the expression into the file of configuration data as "changing" layout, and the layout section is realizable if this is directly used for rear riser hardware. It is actually operating the rear riser hardware constituted according to the expression of the input design section as "operating" the layout section. "Interconnect" is a means in which the reconstruction for passing a logic signal among many chip I/O pins is possible as if the pin interconnected with the wire. One in the inclusion interconnect wire during the crossbar chip in the hierarchy of the partial crossbar between logic chips and crossbar chips is said. [in / partially / in "pass" / crossbar interconnect] A "pass number" specifies specific pass out of many pass which has connected the chip of a pair mutually. "ERCGAs" are prehension of the gate array which can be reconfigurated, i.e., combination logic, and an input/output connection (and additional storage) electrically, and the function and interconnect are constituted and reconfigurated many times by only supplying an electrical signal. A "logic chip" is ERCGA used for realizing combination logic of the input design section in a rear riser system, storage, and interconnect. "L chip" is the memory module attached in the location of a logic chip or a logic chip, or a device module

specified by a user. An "interconnect chip" is the device which can be reconfigurated on the electric target which can perform interconnect of the arbitration between I/O pins. A "routing chip" is an interconnect chip used for direct interconnect or channel routing interconnect. A "crossbar chip" is an interconnect chip used for crossbar interconnect or partial crossbar interconnect. "X chip" is a crossbar chip in the partial crossbar which connects L chip mutually. "Y chip" is a crossbar chip in the 2nd level of hierarchy partial crossbar interconnect, and X chip is connected mutually. "Z chip" is a crossbar chip in the 3rd level of hierarchy partial crossbar interconnect, and Y chip is connected mutually. A "logic board" is the printed circuit board and interconnect chip which transmit logic. A "box" is physical enclosure like a card cage equipped with one or more logic boards. A "rack" is physical enclosure equipped with one or more boxes. "System level interconnect" is interconnecting a bigger device than each chip, and is a logic board, a box, a rack, etc. A "logic cell array" or "LCA" is the specific example of ERCGA, and is Xilink. It is manufactured by Inc. and others and is used for a suitable example. "Logic block which can be constituted", or "CLB" is the small block and flip-flop of logic which can be constituted, and expresses the combination logic and the storage in LCA. "Layout memory" is a memory device which realizes the memory function specified in the input design section. "Vector memory" is a memory device used for catching the reply signal from [many of] the layout section realized by the rear riser system in supply and/or a rear riser system in many stimulus signals. "SUTIMYURETA (stimulator)" is a device in the rear riser system used for supplying each input terminal of the layout section which had the stimulus signal realized. A "sampler" is a device in the rear riser system used for catching the realized reply signal from each output terminal of the layout section. A "host computer" is the computer system of the common use to which the host interface hardware of a rear riser system is connected, and the configuration and actuation of the hardware of a rear riser are controlled. An "EDA system" is electric automated design system, i.e., the system about the tool of the computer base used for creating, editing and analyzing the electrical-design section. In the case of many adapting a rear riser system, a host EDA system generates an input design file.

[0003] If the gate array which has sufficient capacity to hold the single large layout section and which can be reconfigurated is used, many of rear riser technology is unnecessary. However, this is never unrealizable from two reasons. The logic capacity of ERCGA is not [1st] the same as the integrated circuit which cannot reconfigurate the size same on the physical target manufactured using the same manufacturing technology probably. The function for reconstruction takes the remarkable space of a chip. ERCGA has the switching transistor which draws a signal, and the storage transistor for controlling these switches, and the chip which cannot be reconfigurated is equipped with metal trace here. And these transistors can be used for ERCGA as logic. By actual layout, I hear that the resource is not used and it has the regularity needed for the chip which can be reconfigurated. The reason is that regular arrangement and regular routing of logic structure cannot use the available gate 100%. Association of these counting for manufacturing ERCGA is about 1 of logic capacity of chip which cannot be reconfigurated/10. In actual activation, the maximum gate capacitance required of ERCGA is the 9,000 gates for now (Xilinx XC 3090). In the integrated circuit which was manufactured using the same technology and which is used commonly mostly now, more than 100,000 gate logic capacity is given (Motorola). Usually using the integrated circuit of many beyond 10-100, or it for the 2nd, and forming a real digital system on many printed circuit boards is known well. Even if the logic capacity of ERCGA is a case equivalent to a large-scale integrated circuit, although almost all digital systems are realized, many such [still] chips will be used. However, since the logic capacity of ERCGA is not equivalent to a large-scale integrated circuit, further many chips are needed. Finally, in order for a rear riser system to have the logic capacity of a single large-scale chip, the rear riser system needs to have ERCGA of the order of 10. In order for this rear riser system to have the capacity of such a chip, ERCGA of 100 order is needed. This must be noticed about being needed regardless of a special manufacturing technology. By the manufacturing process by doubling the number of the transistors per chip, if the capacity of ERCGA is doubled, the capacity of the chip which cannot be reconfigurated will become twice, therefore the whole layout size will become twice similarly.

[0004] While enabling it to interconnect hundreds of ERCGA(s) by the method of reconfigurating, it is necessary to enable it to change layout into the configuration of hundreds of ERCGA(s) electrically for these reasons, in order to develop an effective rear riser system. this invention does not attain to even the technology of ERCGA itself, and carries out Seki only to the technology which is for developing a rear riser system from many ERCGA(s). ERCGA technology does not show how a rear riser system is developed. The reason is that it is another problem. The ERCGA technology for the interconnect logic element which constitutes the one whole IC chip and which can be reconfigurated cannot apply many to interconnecting. The switching transistor which lets a signal pass in the direction of either can attain ERCGA interconnect easily. Since the one whole chip is covered and an obstruction does not exist, the path of a large number used for interconnect exists. Since the chip is small, the delay of a signal is also small. It is difficult to interconnect many ERCGA(s). The reason is accompanied by the IC package pin and the printed circuit board. I hear that the number of the paths for interconnect is restricted, and the number of the pins which can be used is sometimes restricted. I/O of the signal of a chip must be performed through an active pin buffer, amplifying, for example. An active pin buffer can send a signal only to an one direction. With the interconnect technology of the rear riser system which bigger delay than the delay produced with one chip produces by trace, ERCGA is completely an option and these buffers and circuit boards solve these problems. Finally, it is not necessary to change layout into the configuration of many chips with ERCGA technology. interconnect of a rear riser system completely differs from the interconnect in ERCGA, and determines and constitutes interconnect -- an option is completely needed. ERCGA is developed using the quick and precise silicon technology which can be used for predetermined time amount. (1989 Xilinx XC3000 LCA is developed using 1-micron SRAM technology.) This is the same technology as the quick and precise system realized. ERCGA is general-purpose, and since it has the interconnect which can be reconfigurated, the present gate array and the chip of common use are equipped with the factor which is not precise. A rear riser system repeats the support to the versatility and the reconfigurability of ERCGA higher than level. Therefore, roughly, the rear riser system of the present more precise system always serves as a fixed factor of the order of 1 rather than is more precise. The rear riser system of board level realizes a gate array, and the rear riser system of box level realizes a board and a large-scale common use chip. Furthermore, the rear riser system of rack level realizes a box. Layout structure is strongly influenced of packaging. I/O pin width of face: On VLSI chip level, 100 I/O pins can be developed easily, and although 200 pins are difficult to develop, about 400 pins, it is hardly developed, without not using. On board level, these numeric characters are twice generally. logic density: -- the board is often equipped with five VLSI chips, and is also equipped with ten VLSIs -- possible -- it is also -- it is not common to have 20 VLSIs. The reason is only that the maximum of an actual board is restricted to about 200 square inches. The box is usually equipped with ten to 20 board, and, occasionally is equipped with 40 boards. Interconnect density: When the plane of a two-dimensional wire can be used several sheets, a module interconnects on a chip and a board completely. However, in essence, the back does not necessarily interconnect so completely in box level, case [single dimension-]. Constraint of these packaging affects considerably the system structure seen in a useful rear riser system. By the rear riser system, a single logic chip usually constitutes the only module from the layout section realized by eye backlash which is low density. The complex of the logic chip in one board realizes one or two VLSI chips. The box of a rear riser board realizes a single board in the layout section. Furthermore, the rack of a box realizes the box of the board of the layout section. Therefore, the logic of the board level of a rear riser system and the complex of interconnect need to have I/O pin width of face in the same logic and interconnect capacity list as the VLSI chip of the layout section. The box of a rear riser system needs the same logic and interconnect capacity as a board, and I/O pin width of face of the layout section, and the rack of a rear riser system needs the same logic and interconnect capacity as the box of the layout section.
[0005]
[Embodiment of the Invention] The contents table 1. rear riser hardware system 1.1 logic and interconnect chip technology 1.2 interconnect architecture 1.2.1 Nearest contiguity interconnect

1.2.2 Crossbar interconnect 1.2.3 Interconnect tri-state network 1.2.4 System level interconnect 1.3 The configuration element 1.3.1 of the specific purpose layout section memory 1.3.2 A stimulus and response 1.3.3 user assignment device 1.4 A configuration 1.5 host interface 2. rear riser layout conversion system 2.1 Layout section reader 2.2 Base element conversion 2.3 division 2.4 Application 3.1 of network listing and an interconnect 3. rear riser The rear riser logic simulation system 3.1.1 Transmission system 3.1.2 of logic simulation, a stimulus, and a response Logic simulation, operating kernel 3.1.3 Use 3.1.4 of a rear riser logic simulation system Implementation-izing 3.1.5 of two or more conditions The transmission 3.2 of the condition to other simulations [simulation / about delay of a rear riser / expression 3.1.6 rear riser] rear riser fault simulation system 3.3 rear riser logic simulator evaluation system 3.4 rear riser prototyping system 3.4.1 Realized virtual meter 3.5 Rear riser executive system 3.6 Rear riser production system 3.7 Rear riser computing system 4. suitable example 4.1 Hardware 4.2 Software [0006] 1. RIARAIZA Hardware-System RIARAIZA Hardware System (Drawing 1) -- :11 -- at Least Two Logic Chips (Usually) dozens of pieces or hundreds of pieces, and 2 -- it is additionally like a memory module and the device module specified by a user -- The interconnect to which one-set L chip equipped with the component for one or more specific purposes is connected to all L chips that can interconnect 2 I/O pin and which can be constituted, 3) The host interface connected to all the devices that the host for a host computer, a configuration system and data I/O, or control can use, 4) It has the configuration system connected to the host interface, L chip in which all configurations are possible, and the interconnect device. This hardware is usually mounted with the gestalt of a logic board, a box, and a rack, and it connects with a host computer. This hardware operates under control of a host computer.

[0007] 1.1 Logic and interconnect chip technology 1.1.1 In order for a logic chip device device to serve as a rear riser logic chip, the digital logical circuit equipped with combination logic (and additional storage) the condition [a capacity limit] must be able to constitute :1 device this device of whose must be the gate array (ERCGA) which can be reconfigurated electrically.

2) A device must be able to reconfigurate the function and internal interconnect electrically in the point that it can constitute so that various logical circuits may be suited, any number of times.

3) A device must connect an I/O pin freely with a digital circuit network regardless of a specific network or the I/O pin to specify, and the partial crossbar of a rear riser system or direct interconnect must be able to interconnect a logic chip with the sufficient result. There is a logic cell array (Logic Cell Array (LCA)) as an example of the logic chip in which reconstruction suitable as a logic chip is possible ("The Programmable Gate Array Handbook", Xilinx Inc., San Jose, CA, 1989). This logic chip is Xilinx. It is manufactured by Inc. and others. This chip is equipped with the two-dimensional array of logic block (Cnofigurable Logic Block (CLB)) which can be constituted. This two-dimensional array interconnects by wiring the segment arranged at the row and column between CLB and IOB while being surrounded by the I/O block (IOB) which can be reconfigurated. Each CLB is equipped with some input terminals, the mux input combination logical-circuit network which can reconfigurate a logic function, one or more flip-flops, and one or more output terminals which can be connected by interconnect in which the reconstruction in CLB is possible. Each IOB can be reconfigurated and it can consider as the input buffer or output buffer of a chip. Furthermore, each IOB is connected to an external I/O pin. Interconnect is formed between CLB, IOB, and a segment through the pass transistor and interconnect matrix which can be reconfigurated by connecting the segment which wired to CLB, IOB, and each other. The function in which all reconstruction is possible is controlled by the bit in the serial shift register of a chip. Therefore, LCA is completely constituted by the shift of a "configuration bit pattern." The time amount which a configuration takes is 10 - 100 microseconds. Xilinx LCA of 2000 and 3000 series is equipped with CLB of 64-320, and can use IOB of 56-144. A LCA netlist (netlist) conversion tool (shown below) creates logic to CLB, and makes interconnect between CLB and IOB the optimal. By constituting interconnect between CLB and an I/O pin, as for a specific network or the specific I/O pin to specify, LCA can connect an I/O pin freely with a digital circuit network independently. In order to materialize a

rear riser system suitably, a LCA device is used as the logic chip. As an array of other classes suitable as a logic chip, there is an array which can be reconfigurated to ERA, i.e., an electric target. The device of the type of ERA60K of plessey is one of those are marketed. This consists of loading a configuration bit pattern to RAM partially. ERA is constituted as an array of 2 input NAND gate. According to the value of RAM, each of 2 input NAND gate is interconnected mutually independently. ERA switches connection with a series of interconnect paths of a gate input terminal. ERA 60100 is equipped with about 10,000 NAND gates. The surrounding I/O cel of an array is used for connecting a gate input terminal and/or an output terminal to an external I/O pin. As shown below, an ERA netlist conversion tool outputs a configuration bit pattern file, while it creates logic in the gate and makes interconnect between the gates the optimal. In ERA, by enabling the configuration of interconnect between the gate and an I/O cel, a specific network or the specific I/O pin to specify can connect an I/O pin freely using a digital circuit network independently. As a logic chip which can be used as a logic chip and which can reconfigurate the class of further others, it can eliminate to EEPLD, i.e., an electric target, and there is a programmable logic device ("GAL Handbook", Lattice Semiconductor Corp., Portland, OR, 1986). Lattice generic array logic (Lattice Generic Array Logic (GAL)) is one of commercial things. This consists of loading a bit pattern to the portion of a logic configuration. GAL is constituted as an array of the sum of a product which has an output flip-flop, and the configuration is Xilinx. It does not have versatility rather than LCA. By GAL, an I/O pin did not need to be connected to the logic between all input pins and between all output pins, and requirements are satisfied partially. GAL has 10-20 pins and has comparatively small structure. However, GAL is used as a rear riser logic chip. The programmable details about a logic chip are explained in U.S. Pat. No. 4,642,487, No. 4,700,187, No. 4,796,216, No. 4,722,084, No. 4,724,307, No. 4,758,985, No. 4,768,196, and the No. 4,786,904 specification. Here, the contents of these specifications are used for explanation.

[0008] 1.1.2 The interconnect chip device interconnect chip is equipped with the crossbar chip used for the whole crossbar interconnect and a part, and the routing chip used for direct interconnect and channel routing interconnect. In order for a device to serve as a rear riser interconnect chip, :1 device forms many logic interconnect among the groups of the I/O pin chosen as arbitration immediately, and each interconnect must be able to supply these signals to an output I/O pin while receiving a logic signal from the input I/O pin.

2) A device must be able to be reconfigurated in the point of specifying the interconnect electrically, and it must be able to be re-specified that it can suit many various layout.

3) When it interconnects the tri-state network in partial crossbar interconnect using crossbar addition technology, a device must be able to materialize the addition gate. (In not using crossbar addition technology, it uses other tri-state technology so that a tri-state section may explain.)

The ERCGA device mentioned above, i.e., LCA and ERA, and EEPLD have satisfied these requirements, and are used as an interconnect chip. The function which can constitute almost all the digital circuit network for logic for an interconnect chip most or when not using at all can send data to an output pin from a direct-input pin. LCA is used as a crossbar chip, in case a rear riser system is materialized suitably. TI 74AS A 8840 digital crossbar switch (SN 74 AS 8840 Data Sheet, Texas Instruments, Dallas, TX, 1987), i.e., the crossing switch device usually used for a telephone switch, can be used as an interconnect chip. By the way, when applying these crossbar switch devices to the configuration which changes dynamically during actuation, the reconstruction speed which is equal to data transmission speed is obtained. This reconstruction speed is quicker than the configuration speed of an ERCGA device. Consequently, such crossbar switch devices are more expensive than ERCGA and low capacity, and will create the rear riser interconnect chip which is not not much desirable.

[0009] 1.1.3 An ERCGA configuration software configuration bit pattern is loaded to ERCGA according to user assignment, and constitutes the logic. It is unreal that a user constitutes logic independently. Therefore, a netlist conversion software tool is obtained by usually manufacturing ERCGA equipment. This tool changes into a configuration bit pattern file the logic specification with which the netlist file is equipped. The netlist conversion tool offered by the computer maker of ERCGA is used for rear riser layout conversion system. If rear riser layout conversion system

reads and changes a netlist conversion tool in the layout section, divides into a logic chip and determines interconnect further, the interconnect chip in the netlist and rear riser hardware to each logic will be generated. A netlist file is a list of these interconnect that consists of all base elements (a gate flip-flop and I/O buffer) and a single logic chip, or an interconnect chip. Rear riser layout conversion system supplies an ERCGA netlist conversion tool to each netlist file, and obtains the configuration file of each chip. It uses a suitable tool, in using various devices as a logic chip and an interconnect chip. A configuration file is equipped with a binary bit pattern, and if this is loaded to an ERCGA device, it constitutes a file according to the specification of a netlist file. ERCGA devices are collected to the single binary file used for memorizing these files eternally and constituting the rear riser system of the layout section before actuation. Rear riser layout conversion system is based on the netlist and configuration file format which are specified by the ERCGA computer maker of a tool.

[0010] 1.1.4 Since LCA is used as a logic chip and a crossbar chip in order to embody a netlist conversion tool rear riser system suitably, it is Xilinx. A LCA netlist conversion tool and its file format are explained here. With the LCA netlist conversion tool (XACT) made from Xilinx, while the logical-circuit network of netlist format is given, a logic element is automatically created by CLB. About the location of an I/O pin, a logic element can be constituted from optimal method and internal interconnect can be made easy. Therefore, this tool solves how internal interconnect of a logic chip is constituted, and creates the configuration file as that output. When each LCA is only changed, and a logical-circuit network is too large to suit single LCA, a failure produces a LCA netlist conversion tool. Xilinx A LCA netlist file is called a XNF file. This is an ASCII text file, is equipped with the statement in the one-set XNF file to each base element, and specifies the name of the network connected to a base element, pins, and these pins. These networks interconnect in the LCA netlist and have connected the not a network but LCA base element of the input design section. Although some files in a XNF file support the network of the input design section directly as a result of layout conversion, other files do not correspond. It is the XNF file base element statement for specifying 2 input XOR gate to call. for example, these -- 'I 1781 -- ' -- ** -- The input pin of said 2 input XOR gate is connected to the network called 'DATA0' and 'INVERT'. The output pin I : linked to the network called 'RESULT' -- SYM -- 1781, XORPIN, and O, O, RESULTPIN1, I and DATA0 -- PIN0, I, an INVERTEND input, and an output I/O pin buffer (OBUF for IBUF for an input, and an output) By adding the statement for specifying an I/O pin, it is specified similarly. These are :SYM which is a fundamental statement to OBUF and is made to drive in an I/O pin 'P57' by this through the network which calls 'RESULT' 'RESULT_D', IA_1266, OBUFPIN, O and O, RESULT_DPIN, I and I, RESULTENDEXT, RESULT_D, O, and LOC=P57Xilinx. LCA is called a RBT file. This is an ASCII text file and is equipped with the stream of '0' and '1' which specifies the binary bit pattern used for constituting the header statement which identifies the portion constituted, and the portion for actuation.

[0011] 1.2 interconnect architecture -- when actual, in order to realize the large-scale input design section and to have to use many logic chips, it must connect with the interconnect which can reconfigurate the logic chip of a rear riser. By this interconnect, the signal in the layout section flows between separation logic chips as needed. This interconnect is equipped with association of electric interconnect and/or an interconnect chip. In order to realize the large-scale layout section in a rear riser system, it is total and the logic chip which has the I/O pin of what 10,000 must be supplied by interconnect. Interconnect can certainly constitute the input design section broadly, is still more nearly high-speed and can make delay during a logic chip min while it is economically easy extensible as system size becomes large. The outstanding size and the cost of interconnect should also increase directly the number of averages of the pin of the network unit in the actual layout section as the total number of the logic chip to connect increases, since it is a small number unrelated to the size of the layout section. The number of the specific logic chip capacity and the logic chips to be used and the number of logic chip pins also increase directly as the capacity of the layout section increases. Therefore, the size and cost of interconnect which were excellent with the capacity of the layout section also change directly. : explaining the interconnect structure of two classes -- the 1st section explains

contiguity interconnect and the next section explains crossbar interconnect. The nearest contiguity interconnect is constituted by a logic chip and the interconnect mixed and constituted according to the field of two-dimensional, a three dimension, or the dimension beyond it. In the nearest contiguity interconnect, matrix organization or the printed circuit board of a gate array chip is widened even to organization of a logic chip. The configuration of the predetermined input design section is determined by the arrangement same with being used when developing a chip and a board, and the routing process. Crossbar interconnect differs from the logic chip which interconnects. Crossbar interconnect can be superficially constituted based on multi-input multi-output organization of the crossbar used for transmission and an operation. Although the nearest contiguity interconnect becomes large as logic capacity becomes large, it will become difficult and uncertain for large-scale interconnect to become slowly as a routing path crowds, and to opt for a configuration. Although the mere crossbar is very high-speed because of the substantivity and easy to constitute because of the regularity, it will become impractical magnitude immediately. Although partial crossbar interconnect holds almost all substantivity and the regularity of a mere crossbar, it increased on the direct target with increase of layout section capacity, and has realized ideal rear riser interconnect. A partial crossbar is used for a suitable example although the interconnect except having illustrated can be used in an actual rear riser system. This use is guessed through this whole specification.

[0012] 1.2.1 Nearest contiguity interconnect 1.2.1.1 In direct interconnect direct interconnect, it carries out not using an interconnect chip, and all logic chips are directly connected mutually in a regular array. This interconnect only consists of the electrical installation during a logic chip. Interconnect of a logic chip can form the pattern with which many differ. Generally, the pin of one logic chip is divided for every group. Therefore, the group of each pin is connected to the group of the same pin of other logic chips etc. in all logic chips. Each logic chip is only connected [in / that is, / all the logic chips of a lot, i.e., physical semantics,] only with the nearest contiguity logic chip in the semantics of the topology of an array at least. A logic signal is transmitted to the I/O pin of another side from one I/O pin, without carrying out direct continuation to them, in carrying out direct continuation of these logic chips to other logic chips which achieve the function as an interconnect chip for all the input design section networks that connect logic to one or more logic chips, or connecting with them through a series of logic chips of other, and connecting with either of the logic which a chip realizes. Thus, in addition to sharing of layout section logic, any predetermined logic chips are constituted so that an interconnect signal may be transmitted to the chip of another side from one chip. The tangent was carried out to the pin which connects to an exclusive logic chip pin the non-logic chip resource which cannot perform an interconnection function, and has connected connection or a logic pin mutually around an array, and it has connected with it. In the specific example shown in drawing 2 , it has the logic chip arranged in a line and the two-dimensional grid of a train, and the chip is equipped with the group of four pins connected to the contiguity logic [each] chip which has memory on the north side, on the south, the east side, and the west side, I/O, and the device specified by a user connected on the outskirts. This interconnect is more extensible from the two-dimensional thing explained here to the thing of high order origin. Generally, when making 'n' into a dimensionality, the pin of each logic chip is divided into 2*n groups. Each logic chip is connected with other 2*n logic chips with the regular gestalt. Although other modification is the same, the magnitude of the group of a pin is not equal. Based on the number of logic chips, and the number of each pins, the size and set of pin group size are chosen, and while making into min the number of the logic chips which intervene between the logic chips of 2, he fully interconnects between the chip pairs which adjoin directly respectively, and is trying for a network to lead only to these two chips. In order to determine how the logic chip for interconnect is constituted, it must be determined how the chip for logic is constituted. In order to constitute a logic chip, as the section of :1 base-element conversion described, the logic of the layout section is changed into the base element gestalt of a logic chip.

2) Segment and arrange the logic base element in a logic chip. the sub network which conforms respectively in the logic capacity of a logic chip segments the layout section -- in addition, a sub network is arranged to each other and the amount of interconnect needed is made into min. It

determines how a logic base element is assigned to a logic chip using standard segmentation and the arrangement tool method which are used in a gate array or standard cell chip automatic segmentation, and an arrangement tool ("Gate Station Reference Manual", Mentor Graphics Corp, 1987), and interconnect is attained. This is the method of being acknowledged and omits explanation beyond this here.
3) Wire the interconnect during a logic chip. That is, a specific logic chip and I/O pin interconnect are selected out of a logic chip, it determines how a chip is constituted using a standard routing tool like a gate array or a standard cell chip automatic routing tool ("Gate Station Reference"Manual", Mentor Graphics Corp., and 1987), and interconnect is attained. Since this is the method of being acknowledged, it omits explanation beyond this except for how this method is applied to the problem of interconnect here. The array of a logic chip is dealt with by the single large gate array or the same method as a standard cell chip. The logic subnetwork segmented respectively supports the large-scale gate array logic macro, and the logic chip I/O pin which interconnected has specified the wiring channel used for routing. Especially, it has the pin for every group of a logic chip I/O pin and the channel of the same number which interconnected in each direction of routing. The same channel is used in each end, without restraining routing by many routing layers using the method same with removing channel constraint of a gate array, since many interconnect is possible between logic chips.
4) the criteria adjusted when it was impossible to interconnect because of the excess of routing (when for routing of the channel not to be carried out in a certain point between routing processings) -- using -- the layout section -- re--- segment and/or rearrange, remove an excess and try interconnect again.
5) Change the specification about which network uses which channel into the role of the specific pin to each netlist to a logic chip and logic chip signal according to correspondence between a specific routing channel and an I/O pin. With the specification of a logic base element, I/O pin specification and the specification of the gestalt of the interior interconnect of a logic chip are sent out to the netlist file for every logic chip.
6) While generating the configuration file for every logic chip using a logic chip netlist conversion tool, create the final rear riser configuration file for combination and an input design for these.
[0013] 1.2.1.2 Channel routing interconnect channel routing interconnect is deformation of direct interconnect. In this case, a chip is divided into the interconnect chip which is not used as logic but only interconnects, and the logic chip chiefly used as logic. Especially a logic chip only merely connects an interconnect chip rather than carries out [instead] direct continuation of each other. In addition, in all points, channel routing interconnect is created according to the direct interconnect method. One or more logic chips and a network interconnect by constituting a series of interconnect chips called a routing chip. While a routing chip connects these logic chips, each other is connected and logic-connection is established between logic chip I/O pins. This is used for the 'circuit board' which can be constituted. The logic chip is arranged with the gestalt of a line and a train, and the perimeter is completely surrounded with a routing chip as shown in :, i.e., drawing 3 , which makes channel routing interconnect two-dimensional as an example. All constitute an array from a line which consists of routing chips, and a line which consists of a logic chip and a routing chip by turns by turns. Thus, around the logic chip, the routing chip is arranged without the break in the line writing direction and the direction of a train. The pin of each chip is divided into four groups, i.e., four edges called "the north side, the east side, on the south, and the west side." The pin of each chip is connected to four nearest contiguity chips in the shape of a grid. : That is, connect the pin of the north side with the pin of the chip contiguous to the north side on the south, and connect an east side pin to the west side pin of the chip which adjoins the east side. It is the same as that of the following. This model is extensible even to a bigger dimension than two-dimensional [of the above-mentioned example]. Generally, when making 'n' into a dimensionality, the pin of each logic chip is divided into 2*n groups. Each logic chip is connected to 2*n contiguity chips. At the center of an array, the routing chip of an individual (2**n-1) exists to each logic chip. Based on the feature of a logic chip and a routing chip, generalization of this channel routing model is used similarly. The pin of a logic chip can be divided into some groups. The pin of a routing chip can also be divided

into some groups. However, the group number of the pin of a routing chip does not need to be the same as the number of the pins of a logic chip. The logic chip and the routing chip do not need to be equipped with the pin of the same number. These deformation is applied as long as it is connected only with the limited set of the nearest contiguity chip, no matter it may be the regular array of a logic chip and a routing chip and may be what predetermined logic chip. the interconnect during a logic chip -- a logic chip -- minding -- coming out -- there is nothing, and if it uses for direct interconnect, while determining how a logic chip is constituted [wiring through an interconnect chip] using the same method, it is determined how an interconnect chip is constituted. The logic signal of a network flows through a routing chip required to complete interconnect and the routing chip of the same number. Since propagation of a signal is overdue with each routing chip, as the routing chips with which a signal flows increase in number, the signal propagation time delay through interconnect becomes longer. While segmenting the logic layout section so that the requirement of routing may be made with min, generally it is desirable to arrange each partition for a specific logic chip. the criteria adjusted when routing was not able to interconnect by being superfluous -- using -- the layout section -- re--- it segments and/or rearranges and interconnects again. this cycle is required -- it is restricted and repeated.
[0014] 1.2.2 Crossbar interconnect 1.2.2.1 A perfect crossbar interconnect crossbar is the interconnect arc theque char which can connect a pin with other pins without constraint. In the switching circuit network of a computer and a communication link device, this is widely used, in order to communicate a message. Even if the interconnect constituted as a perfect crossbar is the combination of interconnect of what kind of pin, even if it is what kind of input design and logic chip segmentation, it can attain interconnect directly by interconnect which can be constituted, while connecting with all logic chip pins. even if the reason is what kind of pin -- what kind of -- others -- it is because direct continuation can be carried out to a pin. However, the practical single device which can interconnect many logic chips does not exist. For example, the logic board of a suitable example is equipped with 14 logic chips which have 128 pins which should be connected respectively. It became 1792 pins in total and is over the capacity of a practical single chip far. A crossbar can consist of a practical interconnect chip and a device. By constituting these, interconnect of arbitration is realizable between I/O pins. In crossbar interconnect, these are called a crossbar chip. The general method of constituting crossbar interconnect from a practical crossbar chip interconnects using one crossbar chip with the pin and other logic chip pins of the same number by which a crossbar chip has one logic chip pin. Drawing 4 shows an example simplified extremely, in order to make it intelligible. Four logic chips which have eight pins respectively are interconnected. The crossbar chip which has nine pins respectively is used. The pin H of the logic chip 4 is connected with the pin of the logic chips 1, 2, and 3 by the train by the side of the three leftmost of a crossbar chip. Pin G etc. is connected to the pin G of the logic chip 4 by the following train. Since it is connectable inside about the same logic chip, it is not necessary to connect a pin and other pins. Eight trains which a crossbar chip adjoins interconnect the logic chip 3 and the logic chips 1 and 2. The logic chip 4 is not included. The reason is that it has connected the pin of the logic chip 4 to the pin of the logic chip 3 by the first eight trains of a crossbar chip. The last eight trains interconnect the logic chips 1 and 2. A total of 48 crossbar chips are used. Two networks based on an input design show the condition of having interconnected. Network A is driven by the pin D of the logic chip 1, and is received by the pin B of the logic chip 4. The crossbar chip shown by 1 has connected both these pins, it receives from the pin D of the logic chip 1, and what received is transmitted to the pin B of a chip 4. Thus, logic connection is constituted. Network B is driven by the pin F of the logic chip 2, and is received by the pin G of the logic chip 3, and the pin G of the logic chip 4. The crossbar chip 2 performs the 1st interconnect and the crossbar chip 3 performs the 2nd interconnect. Generally, the number of the crossbar chips needed can be predicted. When the crossbar chip which L logic chips which have a P1 piece pin respectively exist, and enables it to connect one logic chip pin with many other logic chip pins respectively as much as possible is equipped with the pin of Px individual, one pin in :1 logic chip 1 must be connected to the P(L-1) 1 piece pin of the logic chip from 2 to L. The crossbar chip of P(L-1) 1-/(Px-1) individual is needed for this. In order to connect all pins, the crossbar chip of P(L-1)

12-/(Px-1) individual is needed.
2) Each pin of the logic chip 2 must be connected to the P(L-2) 1 piece pin of the logic chip from 3 to L. The crossbar chip of P(L-2) 12-/(Px-1) individual is needed for this.
3) Each pin of the logic chip L-1 must be connected to the pin ofP1 L logic chip. The crossbar chip of P12-/(Px-1) individual is needed for this.
4) X=(L-1) P12-/(Px-1)+(L-2) P12/(Px-1)+ .... +P12/(Px-1)
=(L2-L)P12/2(Px-1)
Several X of a crossbar chip increases as what multiplied by the square of the number of logic chips and the square of the number of the pins for every logic chip increases. The logic board (14 logic chips which have 128 pins respectively) of a suitable example needs 11648 crossbar chips or 23296 crossbar chips which have 65 pins respectively which have 129 pins respectively. In order to use for a useful rear riser system, crossbar interconnect is large-scale and expensive, and impractical.
[0015] 1.2.2.2 By recognizing that the number of the layout networks which should carry out perfect crossbar network interconnect interconnect can never exceed 1/2 which is the total number of a logic chip pin, magnitude of crossbar interconnect can be made small. Crossbar network interconnect is constituted by two crossbars in logic, it has connected with the one-set connection network which calls all logic chip pins an interchange-circuit network (ICN), and the each gives the number to one half of the totals of a logic chip pin. Since the crossbar chip which connects an one-set logic chip pin to one-set ICN can also return connection to these pins from one-set ICN (withdrawal of the generality of an interconnect chip), this interconnect can also consist of crossbar chips. Each crossbar chip has connected the one-set logic chip pin with one-set ICN. Drawing 5 is drawing showing an example which interconnected the four same logic chips as what was shown by drawing 4 . 16 ICN(s) are prepared using the crossbar chip which has eight pins respectively. Each of 32 crossbar chips connects four logic chip pins using four ICN(s). It constitutes so that it may interconnect with the crossbar chip 1, Network A may be received from the pin D of the logic chip 1 and what received may be transmitted to ICN. Moreover, it interconnects with the crossbar chip 2, Network A is received from said ICN, and the pin B of the logic chip 4 is driven. Thus, logic connection is established. Network B is received by the pin G of the logic chip 4 through the crossbar chip 5 while it drives by the pin F of the logic chip 2 and being received by the pin G of the logic chip 3 through the crossbar chip 4. Crossbar network interconnect of the logic board (14 logic chips which have 128 pins respectively) of a suitable example needs 392 crossbar chips which have 128 pins respectively, or 1568 crossbar chips which have 64 pins respectively. There are few crossbar chips used in crossbar network interconnect than a mere crossbar. The magnitude of crossbar network interconnect becomes large as the product of the number of logic chips and the total of a logic chip pin increases. This reaches the square of the number of logic chips. this is superior to a pure crossbar -- **** -- it is not the direct scaling still desired.
[0016] 1.2.2.3 The additional flexibility which cannot develop a crossbar can be offered by partial crossbar interconnect logic chip itself. The reason is that it can constitute the predetermined input or predetermined output of a logical-circuit network so that any I/O pins can be used. That is, it is because it constitutes regardless of a specific network. Partial crossbar interconnect can be carried out with this flexibility. This is a reason for specifying flexibility during the definition of a logic chip. In partial crossbar interconnect, the I/O pin of each logic chip is divided into a suitable subset the same with dividing each logic chip. The pin of each crossbar chip is connected to the subset of the same pin from each of each logic chip. Thus, a crossbar chip 'n' is connected to the subset 'n' of the pin of each logic chip. The crossbar chip of a subset and the same number is used. Each crossbar chip has the pin of the number which multiplied by the number of the pins of a subset, and the number of logic chips, and the same number. It interconnects with the wire which calls each logic chip / crossbar chip pair the pin in each subset, and the pass of the same number. Since each crossbar chip is connected to the same subset as the pin of each logic chip, the interconnect to the I/O pin in another subset in the pin of another logic chip in one subset in the pin of one logic chip from an I/O pin cannot be constituted. From the same subset of the pin of each logic chip which should interconnect, using

an I/O pin, this interconnects each network and is avoided by constituting a logic chip suitably. Since it enables it to constitute a logic chip no matter what pin [which can be assigned to the logic chip constituted in the logic chip connected to a network / I/O] it may use, one I/O pin is the same as the I/O pin of another side. A common pattern is shown in drawing 6 . In this drawing, each Rhine which has connected the logic chip and the crossbar chip shows the subset of a logic chip pin. Each crossbar chip is connected to the subset of the pin of all logic chips. Conversely, if it says, this shows that each logic chip is connected to the subset of the pin of all crossbar chips. In these examples, the number of crossbar chips does not need to be equal to the number of logic chips. Such a thing cannot be said in the suitable example of implementation. Drawing 7 shows the example which interconnects the four same logic chips as drawing 1 and drawing 2 . Four crossbar chips which have eight pins respectively are used. Each crossbar chip has connected the two same pins in each logic chip. The crossbar chip 1 is connected to each pins A and B of the logic chips 1-4. While connecting the crossbar chip 2 to all the pins C and D and connecting the crossbar chip 3 to all the pins E and F, the crossbar chip 4 is connected to all the pins G and H. In the layout network A of said example, although reception is performed in the pin B of the logic chip 4, the crossbar chip which can interconnect in a driver [in / for Network A / the pin D of the logic chip 1] is not prepared. Since it can assign the logic constituted in the logic chip 4 which receives Network A no matter it may be what I/O pin, Pin C is the same as Pin B, and can use this for other networks. As a result, it is instead received by Pin C, and Network A is constituting the crossbar chip 2, and attains interconnect. Although the layout network B is received by the pin G of the logic chip 3, and the pin G of the logic chip 4, the driver in the pin F of the logic chip 2 and the crossbar chip which can interconnect have not formed this network B. Network B is instead driven by Pin H, and interconnect is attained with constituting the crossbar chip 4. Partial crossbar interconnect is used in the suitable example. This logic board is equipped with 14 logic chips which have 128 pins respectively, and interconnects with 32 crossbar chips which have 56 pins respectively. While dividing a logic chip pin into 32 suitable subsets which have four pins respectively, the pin of each crossbar chip is divided into 14 subsets which consist of four pins respectively. A crossbar chip 'n' is connected to the subset 'n' of each logic chip pin, and each logic chip / crossbar chip pair are interconnected with four pass. The number of the crossbar chips which a partial crossbar uses in all crossbar interconnect is min. The size of a partial crossbar increases directly as the total of a logic chip pin increases. This is [the number of logic chips, and] a desirable result directly in relation to logic capacity further. It is comparatively easy to use this. The reason has a regular partial crossbar, it is possible to express the pass on a table, and it is because the optimal pair of pass is only looked for on a table how further specific interconnect is determined.

[0017] 1.2.2.4 Functional division crossbar interconnect of partial interconnect cannot process the network of that a perfect crossbar can be processed and the same number. In a source logic chip, when the pass with which the I/O pin which is not used only one to other networks results in a destination logic chip is connected with the crossbar chip currently used altogether similarly, partial crossbar interconnect cannot interconnect a network. Although the destination logic chip has the available pin, in such a case, the I/O pin is connected with other crossbars with which all source pins are used, and there is no way which returns from these crossbars to the beginning. It depends for the capacity of partial crossbar interconnect on the architecture. In one extreme example, only one logic chip pin subset exists and one crossbar acts on all pins. Although such equipment has the maximum interconnect capacity, it is unreal perfect crossbar connection. In other extreme examples, subset size has the pin of a logic chip, and the crossbar chip of the same number. Although the capacity for this to interconnect all partial crossbars is min, it has still sufficient capacity. Between extreme examples, each crossbar chip serves as architecture which acts on the pin of 2, 3, or each logic chip beyond it. the number of crossbar chips decreases and the number of pins for every crossbar chip increases -- it is alike, it takes and more interconnect capacity becomes available. Although this modification is observed for some time, in order that various crossbar chips may act, it is because the intact logic chip which cannot interconnect exists. It stops producing such modification generally as width of face becomes large, while the number of crossbar chips decreases more. A perfect crossbar can

interconnect also to whether all pins are defined and the becoming pattern. It is assumed that it is that in which three logic chips which have three pins respectively, and which attached reference numbers 1, 2, and 3 exist as other easy examples, and four networks A, B, C, and D exist. Network A connects the logic chips 1 and 2, Network B connects the logic chips 1 and 3, Network C connects the logic chips 2 and 3, and Network D connects the logic chips 1 and 2. In drawing 8 a and 8b, the pin of each logic chip is shown as a line of a cel, and each crossbar chip covers the train of the number of pins on which a crossbar chip acts, and the same number. In the 1st case (drawing 8 a), three crossbar chips shown with the reference numbers 1, 2, and 3 which have the width of face of one pin respectively are used. Each crossbar chip can connect only one network. That is, the crossbar chip 1 is programmed to interconnect Network A, the crossbar chip 2 connects Network B and crossing BACHITSU 3 connects Network C. Although an intact logic chip pin can also be used, Network D is still un-connecting. In the 2nd case (drawing 8 b), the perfect crossbar which has pin width of face of three pieces can be used instead of the crossbar chips 1, 2, and 3, and Network D can be connected. An analysis and computer-model-ization are performed based on the number of the input design networks which can interconnect by various partial crossbar interconnect architecture. As a result, the narrow partial crossbar is effective for the almost same degree as a wide thing or a wide perfect crossbar. For example, the interconnect used for the logic board of the example of suitable implementation (14 128 pin-logic chips, 32 56 pin crossbar chips) shows 98% of the interconnect capacity which a perfect crossbar has. It is very rare that the actual input design section needs the number of an available multi-logic chip circuit network and logic chip pins for the maximum as assumed in modeling. The actual layout section has networks fewer almost always than the maximum, and has networks fewer than the average number of the network connected by the partial crossbar of an above-mentioned model, and usually quite few networks. This is secured by using many the logic chip pins and crossbar chips of a small ratio rather than absolutely required to hold logic capacity, and it secures that it can carry out in this way, and the actual layout section can almost always interconnect with a narrow partial crossbar. A narrow crossbar chip is quite small and, so, is not more expensive than a wide crossbar chip per pin.

[0018] 1.2.3 I hear that active interconnect is nondirectional and the material difference of active interconnect like interconnect tri-state network partial crossbar interconnect and passive interconnect like an actual wire has it. Each interconnect is actually equipped with a series of drivers and receivers which join together by the metal and trace on a chip boundary. The usual network has one driver and is created using the driver and receiver which were fixed by active interconnect. Some of actually designed networks are tri-state, and they have some tri-state drivers as shown in drawing 9 . In the predetermined time of arbitration, one driver is an active state at the maximum, and other drivers are in the condition of a high impedance to a network. (if a propagation delay is disregarded) All receivers are always in the same logic level.

[0019] 1.2.3.1 When segmenting all the networks that transpose a tri-state network to the sum of a product to the same logic chip, a network can be replaced by 2 State addition of a product, i.e., an equivalent multiplexer as shows by drawing 10 . When active enabling does not exist, this network outputs a low logic level. A tri-state network is sometimes passively made into a high logic level. When it cannot enable by reversing the last addition gate output while reversing the data input to each AND gate if required, the sum of a product outputs a high logic level. When one or more enabling are active, a result serves as addition (OR) of all input signals. This is admitted. The reason is that it has not specified the motion of an actual tri-state driver, when enabling one or more by different data. Drawing 11 a and 11b show "the floating low (floating low), two kinds of network: (floating high), i.e., a "floating high", and,." The amount of [of the layout conversion system of a rear riser system] base element transducer performs replacement of the sum or a product. The reason is Xilinx used as the logic chip and crossbar chip of the example of suitable implementation. It is because LCA is not maintaining the tri-state drive in all networks uniformly. All the I/O pins in the boundary of LCA can be used for a tri-state driver. The number of the tri-state drivers which can be used inside the XC3000 series LCA is restricted, and since the number of the internal interconnect which has connected between chips is small, each driver acts only on one line of CLB. By creating a tri-state network

to these interconnect, it will join division in other constraint and the flexibility of the arrangement of CLB in LCA will be restrained. It is that it is common to coincidence in a certain kind of gate array library cel to make tri-state connection with a small number of driver for every network. The sum of a product is replaced when becoming complicated in this way can be avoided as a result. When continuing and dividing a tri-state network into two or more logic chips by dividing the layout section into a multiplex logic chip, connections between a logic chip and a network are reduced to the single driver and/or single receiver in a logic chip boundary, using the sum of a product locally. Drawing 12 shows two drivers and two receivers together. Two drivers are constituted by the sum of a local product, thus give total of a product by making only single driver connection requirements. Similarly, single receiver connection is continued and constituted to two receivers. Thus, active interconnect is made. In any predetermined points in a tri-state network, it is dependent on which driver a drive "a direction" makes an active state. Although a difference does not have this in any way with passive interconnect, it must constitute active interconnect from active interconnect so that drive and reception may be actively performed in the direction of the right. Depending on a configuration, partial crossbar interconnect can attain this.

[0020] 1.2.3.2 The configuration of three logic addition configurations is based on reducing a network to the sum of a product. A logic addition configuration arranges the addition OR gate in a related logic chip as shown in drawing 13 . The AND gate which generates a product consists of drive logic chips. Each of this drive chip needs the output chip. Each receive logic chip needs an input pin, and when special, an addition logic chip needs the input pin and output pin for each drivers. All of these connection are nondirectional and it is equipped with the OBUF/IBUF pair covering the boundary of each chip. Since the pin of a driver is expensive, it is necessary to choose a drive logic chip as an addition chip. Since it is easy, all related LCA base elements are not shown all over drawing. The actual pass from a driving-input pin to a reception output pin is equipped with CLB of a driver and OBUF, IBUF/OBUF of a crossbar, IBUF of an addition chip, CLB and OBUF, other IBUF/OBUF of a crossbar, and a receiver's IBUF. When crossbar IBUF delay is set to Ix and logic CLB delay is made into C1 grade, all data path delay is C1+O1+Ix+Ox+I1+C1+O1+Ix+Ox+I1. When special (i.e., when the logic chip was set to XC 3090-70 and a crossbar is set to XC 2018-70), the max of the grand total of delay is equal to what added internal LCA interconnect delay to 82ns. The same delay is applied also to enabling. When it interconnects a n bit bus, all enabling are same to each bit of a bus. this special configuration -- setting -- under a drive chip -- the gate of a product -- preparing -- enabling -- the interior -- preparing -- a pin required for a bus -- the number of pins in 1 bit -- they may be n times exactly.

[0021] 1.2.3.3 Arrange the addition OR gate for a crossbar chip in a crossbar addition configuration crossbar addition configuration. In this case, the crossbar chip of some examples is materialized using ERCGA like LCA which can use logic as shown in drawing 14 . Each logic chip needs one pin as one pin and/or receiver as a driver. The crossbar chip needs to have one or more logic elements for the addition gate. Crossbar addition is not performing using all the logic in a logic chip, not using logic in a crossbar chip at all. I hear that the logic arranged at the crossbar chip is not a part of layout logic realized, and a material-difference point has it. Logic only plays the role which attains the interconnection function of a tri-state network. When two or more drive logic chips are prepared with this configuration, there are few pins used conventionally. A n bit bus acts also n times of a pin. All delay is reduced at C1+O1+Ix+Cx+Ox+I1, i.e., a maximum of 51ns. Also enabling has the same delay.

[0022] 1.2.3.4 The addition gate of a crossbar chip is connected through the bidirection connection in a bidirection crossbar configuration as shown by the bidirection crossbar addition block diagram 15 . The AND gate which can enable the pass to the OR gate is prepared into a crossbar chip, and feedback latch rise pass is blocked. Only in the case of a receiver, a logic chip needs one pin, and, in the case of both a driver or a receiver, and a driver, needs two pins. One side of these two pins is a thing for the signal itself, another side is a thing for an enabling output, and this is used for a crossbar chip. Interconnect can be decreased by multi-bit bus using signal enabling [of 1 bits or more]. When it interconnects a bus 1 bits or more through the

same crossbar chip, it is necessary to supply an one-set enable signal to a chip. All data path delay is made into O1+Ix+Cx+Ox+I1, i.e., 42ns, in the suitable example of LCA. When the sum of a product uses two or more CLB(s), additional Cx (10ns) can be added. Enabling delay is dependent on the enabling delay E1 of OBUFZ instead of the output delay O1.
[0023] 1.2.3.5 All the configurations explained until now [bidirection crossbar tri-state configuration] must be noticed about the ability to be used by the same hardware. Only arrangement and interconnect of a base element change. Finally, when a crossbar chip maintains internal tri-state, as drawing 16 shows, the actual tri-state network inside a crossbar chip becomes a duplex by the bidirection crossbar tri-state configuration. The actual tri-state driver of each logic chip is transmitted to the bus of a crossbar chip as it is. This should be attained by interconnect of an enable signal. When not enabling the driver, the bus of a crossbar chip is made to drive. When using LCA as a crossbar chip, the above-mentioned internal tri-state interconnect is used. Especially, TBUF is prepared in the boundary of a logic chip on an IBUF/OBUFZ pair and a crossbar chip boundary at other IBUF/OBUFZ pairs of each logic chip, and each logic chip, and an internal tri-state line is driven. Each enabling passes OBUF and IBUF. The grand total of the data path delay which it enabled is O1+Ix+Tx+Ox+I1, i.e., 39ns, (XC3030-70LCA crossbar), and the grand total of enabling delay is O1+Ix+TEx+Ox+I1, i.e., 45ns. Like [former], when it interconnects a bus 2 bits or more through the same crossbar chip, it is necessary to supply only an one-set enable signal to a chip. It is necessary to set a crossbar chip to ERCGA which has LCA or an internal tri-state function, and contingent [on use of these interior interconnect] with this configuration. It has XC3000 parts, although especially XC2000 series does not have internal tri-state. XC3030 has 80 I/O pins, 100 CLB(s), and the interior of 20 pieces 'a long line' that can be tri-state driven. Thus, such 20 tri-state networks can be interconnected with one crossbar chip under this configuration at the maximum. Although this can serve as a limit of interconnect, it does not pass over it in a part of mere various cases, but it gives the limit of an I/O pin. now -- XC3030 -- 2 of XC2018 -- double -- it is expensive. When using a tri-state configuration for hardware, other configurations do not serve as hindrance but can be used similarly.
[0024] 1.2.3.6 The chart of ****** of all configurations summarizes a configuration.

	ロジック加算	クロスバー加算	双方向性クロスバー加算	双方向性クロスバートライステート
ピン／ロジックチップ双方向性	＝駆動＋受信	2	１データパス１共有可能イネーブル	1データパス１共有可能イネーブル
駆動のみ	第１チップ：０その他：２	1	１データパス	１データパス
受信のみ	第１非加算：２その他：１	1	１共有可能イネーブル	１共有イネーブル

遅延

（ＬＣＡクロスバーチップ：＋ＬＣＡ相互接続、７０ＭＨｚ　ＬＣＡチップスピードであると仮定する）

データパス	８２ｎｓ	５１	４２	３９
イネーブル	８２	５１	４６	４５

チップ毎のリソース

（ｄはドライバの数）

駆動のみ	ＡＮＤ中に２個	ＡＮＤ中に２個	０	０
受信のみ	０	０	０	０
双方向性	ＡＮＤ中に２個	ＡＮＤ中に２個	０	０
クロスバー	０	ＯＲ中にｄ個	ＯＲ中にｄ個	ｄ個のＴＢＵＦ

Clearly, a logic addition configuration is not effective. It is quite a high speed, and crossbar addition uses a small number of pin, and, in many cases, is simple. Although bidirection crossbar addition is still slightly high-speed and has a possibility of decreasing the number of pins of a bidirectional bus, it is quite complicated, and it needs more the logic resource restricted to the crossbar chip. Although the same pin is needed and delay arises by the tri-state configuration, a more expensive crossbar chip is needed.

[0025] 1.2.3.7 It is useful that ****** of an ordinary crossbar addition configuration and a bidirection crossbar addition configuration also tests the property of an effective configuration. The following tables show the number of Crossbars CLB and crossbar CLB delay which are produced when it usually reaches, and many bilateral circuit networks are interconnected using the crossbar addition configuration of bidirection and it uses LCA as a crossbar chip. It has 72 I/O pins and it is assumed that it is the thing using a XC2018-70 crossbar chip which can use 100 CLB(s). Each CLB supports the input terminal to four pieces, and the output terminal to two pieces. Each logic chip does not share enabling, but has connection of a network and bidirection, and assumes in each test that it is a thing using all 72 I/O pins of a crossbar chip.

	クロスバー	双方向性クロスバー
	加算	加算
１８個の双方向性回路網サービング	９ ＣＬＢｓ	１８ ＣＬＢｓ
２個のロジックチップの各々	１ Ｃｘ	１ Ｃｘ
１２個の双方向性回路網サービング	１２ ＣＬＢｓ	２４ ＣＬＢｓ
３個のロジックチップの各々	１ Ｃｘ	２ Ｃｘ
９個の双方向性回路網サービング	９ ＣＬＢｓ	２７ ＣＬＢｓ
４個のロジックチップの各々	１ Ｃｘ	２ Ｃｘ
６個の双方向性回路網サービング	１２ ＣＬＢｓ	２４ ＣＬＢｓ
６個のロジックチップの各々	２ Ｃｘ	２ Ｃｘ
３個の双方向性回路網サービング	１２ ＣＬＢｓ	３０ ＣＬＢｓ
１２個のロジックチップの各々	２ Ｃｘ	３ Ｃｘ

A bidirectional crossbar addition configuration uses CLB up to 2.5 times, and a possibility, i.e., a possibility that internal interconnect delay will become large, that a crossbar chip will not carry out the root increases. However, it is far by the time 100 CLB(s) become still usable. Although a logic chip is instead prepared in a suitable location to operate the special gate with a nondirectional configuration, quite many gates are established in the logic chip. With a bidirection configuration, special Cx delay will often arise and will offset the advantage of the speed. In the suitable example of a rear riser system, the crossbar addition configuration for a tri-state network is used.

[0026] 1.2.4 As for package-izing of the one-set logic chip which interconnected with a system level interconnect crossbar chip, it is common to carry out in a single circuit board. When a system is too large-scale and does not suit a single board, a board must be similarly interconnected using system level interconnect. It is very impractical to extend the single partial crossbar interconnect covering two or more circuit boards and a logic chip for pass wiring of a wide area. For example, the complex of 32 128 pin-logic chips and the crossbar chip of 64 pins is assumed to be what is divided into 16 logic chips and 32 crossbars, respectively on two boards. When cutting complex between a logic chip and a crossbar chip, the grand total needs to perform 4096 interconnect between a logic chip and a crossbar chip through the pair of a back connection. The pass (16 logic x64 piece pins = 1024) which this is an option and is 'middle', namely, connects the logic chip of a board 1 to the crossbar of a board 2 when 16 logic chips and 32 crossbar chips cut for every board, and all the pass (it is 2048 by another 1024 and sum total) of the reverse must be made to cross. In such single interconnect, there is other constraint that there is no possibility of development. According to the definition, each crossbar chip is connected with all logic chips. When it constitutes from a logic chip of the number of specification, more than it cannot be added. Instead, on the circuit board, the complex of the maximum scale of the logic chip and crossbar chip which can be package-ized together is called a logic board, it uses as a module, and these large number are connected by system level interconnect. In order to offer the interchange-circuit network which attains to two or more boards, additional connection which is separated from a board is made to each additional I/O pin of the crossbar chip of each logic board, and a logic board I/O pin is established (drawing 17). The crossbar chip I/O pin of the thing linked to the logic chip I/O pin of a board used for connecting with a logic board I/O pin is another.

[0027] 1.2.4.1 With one means for interconnecting a partial crossbar system level interconnect logic board, while applying partial crossbar interconnect again and dealing with each board like a logic chip, interconnect the I/O pin of a board using the set of an additional crossbar chip. This partial crossbar interconnects all the boards in a box. The 3rd interconnect is again applied to interconnecting all the boxes in a rack etc. By applying the same interconnect method from beginning to end, advantages, such as simplification of a concept and unification with board level interconnect, are acquired. In order to distinguish the crossbar chip in a rear riser system, the partial crossbar interconnect which interconnects the logic chip is called X level interconnect, and the crossbar chip is called X chip. The interconnect which interconnects the logic board is

called Y level interconnect, and the crossbar chip is called Y chip. In X level interconnect, the I/O pin of each logic board is divided into a suitable subset using division of each logic board and the same division. The pin of Y chips each is connected to the same subset of the pin from each of all logic boards. Y chip of a subset and the same number is used. Y chips each have the pin of the result of having multiplied by the number of pins of a subset, and the number of pins of a logic board, and the same number. Similarly additional connection which is distant from a box at the additional I/O pin of Y chips each is made, and a box I/O pin is constituted. This each is divided into a suitable subset using the same division method as the division in each box (drawing 18). The pin of Z chips each is connected to the same subset of the pin from each box. Z chip of a subset and the same number is used. Z chips each have the result of the number of pins of a subset, and the number of boxes by which it multiplied, and the pin of the same number. The method of constituting the additional level of partial crossbar interconnect is continued as long as it is required. When segmenting the input design section, it turns out that the network currently wired by separating from the board top and the board minds the I/O pin of a board, and limitation of the number of the I/O pins of a board has fixed constraint as the number of the I/O pins of a logic chip is limited. In a multiplex box rear riser system, it turns out that the number of box I/O pins is limited etc. The interconnect symmetry means for making the optimal arrangement about a chip, a board, or a cartridge except for the case where a special function like layout memory accompanies is not necessarily required. Shape is taken using one of the methods which explained the bilateral circuit network and the bus in a tri-state section like the crossbar addition method. This method is continued and applied to each level of the interconnect hierarchy with whom the network is connected. The suitable example is as follows.
- Use partial crossbar interconnect hierarchical in three level covering all hardware systems.
- The logic board is equipped with X level partial crossbar which consists of a maximum of 14 logic chips and 32 X chips which have 128 I/O pins which interconnected respectively. X chips each were equipped with eight pass connected to each of four pass respectively connected to 14 L chips (a total of 56 pieces), and two Y chips, and are equipped with 512 logic board I/O pins for every board as a whole.
- One box is equipped with 1-8 boards which have 512 I/O pins which interconnected respectively, and the Y level partial crossbar which consists of 64 Y chips. Y chips each are equipped with eight pass connected to eight pass and one Z chip which have led to X chip of each board through the logic board I/O pin, and are equipped with 512 box I/O pins per box in total.
- A latch has 512 interconnect I/O pins respectively. It has 1-8 boxes and Z level crossbar which consists of 64 Z chips. Z chips each are equipped with eight pass connected to Y chip in each box through the box I/O pin.

[0028] 1.2.4.2 Other methods about the system level interconnect of a logic board which uses the back of a bidirectional bus are needed for activation of bidirection bus system level interconnect computer hardware. Like before, an I/O pin is prepared in each logic board, and the I/O pin of each board is connected to the same I/O pin of all other boards with a bus wire (drawing 19). Some logic board I/O pins are useless. That is, interconnect is impossible to a layout network. The reason is that it carries out by the ability not using the pin connected to the bus wire of all the boards of the others which are sharing the bus by using the bus wire for interconnecting one layout network. The maximum number of the layout network which can interconnect is equal to the number of bus wires, i.e., the number of the I/O pins for every board. When special, in eight boards, 512 bus wires which have connected 512 I/O pins of each board are sharing [******] one common interconnect bus (drawing 20). When it assumes that it is the wiring with which the networks of the 2nd, the 3rd, the 4th, the 5th, the 6th, and the 7th board [8th] differ, in each case, the number of averages of the network linked to each board is 512, and analysis shows that a board and a bus should be permitted to pin width of face of 1166 pieces in all networks. This is partially mitigated by continuing making the number of boards of the single back small. However, the bidirectional bus of a lot and the maximum number of the board which interconnected are restricted. In order to constitute a large system more effectively, the group of a bus is interconnected hierarchical. In the 1st example shown in

drawing 21 , it has 2 sets of buses X0 and X1 which have connected four boards respectively. The bus of X level is interconnected by other bus Y. Each wire in X bus is connected to one of the two of Y by the bidirection transceiver which can be reconfigurated. It is determined whether the wire of X and Y is insulated and whether by the configuration of a bidirection transceiver, X drives Y and a drive or Y drives X. When a network connects only the group of a left-hand side board, or the group of a right-hand side board, only one side or another side of X level bus is used. When both sides are equipped with the board, the wire of X0 and X1 is used respectively, and these wires are interconnected with the wire of Y through a transceiver. Each board needs to have the I/O pin of one width of face and same number of X level bus. When interconnect through Y drives by either bidirection, X0 [i.e.,], or X1, an additional signal flows from X0 and X1, and controls the directivity of a transceiver dynamically. The function which analyzes this interconnect and interconnects the network between boards is shown. Under the present circumstances, it is assumed that they are the same number of network pins as the above, and the number of I/O pins. Although the same width of face as the grand total of all networks is needed by the single level method, this is divided into two and the maximum width needed is decreased to 10% - 15%. It is only that a hierarchy has a board two free [per bus], or two groups' board at the maximum (drawing 22). Bidirectional-bus interconnect is simple, and although it is easy an assembly, it is expensive. The reason is that it makes it useless by connecting quite many logic board I/O pins to the network of other boards. In order to avoid this, even if it introduces hierarchization and the short circuit back, it is proved that an effect is very small. Furthermore, single level back bus interconnect will remove the speed in dominance, and the advantage in the field of cost from a partial crossbar by introducing a bidirection transceiver. As a result, a partial crossbar is used for system level interconnect of a suitable example.

[0029] 1.3 Although the configuration element for the purpose of [of the specific purpose] configuration element specification is a hardware configuration element which realizes an input design and is attached in the location of L chip of the logic board of a suitable example, it is not the combination logic gate or flip-flop which constitutes a logic chip.

[0030] 1.3.1 The input design section of layout section memory many is equipped with memory. It is ideal if the logic chip is equipped with memory. The current logic chip device is not equipped with memory. Still needing the main memory of a megabyte scale, supposing it has memory, this is never being able to expect a logic chip. Therefore, suppose that a layout memory device is prepared into a rear riser system.

[0031] 1.3.1.1 Constitute the architecture of the memory architecture layout section memory module of the layout section based on the following requirements. : Since the memory module of a layout section is a part of layout section, it needs to enable it to interconnect in other components and freedom.

b) It is necessary to prepare flexibility in assignment of data, the address, and control I/O, and to interconnect pass so that effective interconnect can be performed like a logic chip.

c) It is necessary to enable modification of a configuration of that one or more layout section memory which has various capacity and bit width of face is realizable.

d) A host interface needs to be accessible so that the layout section and debugger type dialogue can be performed.

e) A memory module does not need to be dynamic and needs to be static. The layout section can be made to run by the clock speed of a stop, a start, or arbitration at will by this. The general architecture of a memory module which satisfies these requirements is shown in drawing 23 .

In order to maintain an interconnect possibility with the layout section, and the flexibility about the physical configuration of a rear riser system, a memory module is designed so that it may connect with L chip socket connected to the same interconnect and other same pins as the replaced logic chip with a plug. Only a required module is attached. Direct continuation of the RAM chip is not carried out to interconnect. The reason is that it has mainly set the data, the address, and the control function of a chip to the specific pin. Since a success of partial crossbar interconnect is dependent on the function of the logic chip which can assign internal interconnect with an I/O pin freely, the non logic chip device arranged in the location of a logic

chip needs to have the same function. While attaining this, in order to provide a memory module with other logic functions, a logic chip is interconnected with installation and a RAM chip is interconnected with X chip of a crossbar in a memory module. A specific RAM pin is interconnected with X chip pin chosen as arbitration, and a memory module is constituted. Under the present circumstances, the same L-X pass as a logic chip using it for the location which uses a memory module is used. Many logic chips are used for every module rather than one piece. The reason is that there are many RAM pins which should be connected, and L-X pass. A memory module is provided with a configuration possibility and host accessibility with the logic chip of a memory module. The address, data, and control pass are constituted so that the RAM chip which has various capacity, bit width of face, and input/output structure may be connected through a logic chip. A memory module can consist of one large-scale memory or some small-scale memory. While connecting each of these logic chips to a host interface bus, a host processor realizes the function which can access RAM at random by constituting a bus interface logic in a logic chip. By this, a computer program like a debugger is used, and a memory content is inspected and corrected. The example of such logic structures is shown below. The available, high-density, and cheap static memory with which are satisfied of the requirements about the timing of the layout section to realize is chosen as layout section memory. Such a device is set to CMOSSRAM of 8 bit 32K like FUJITSU MB84256 in the suitable example. According to this, speed can be reduced to 50ns. A return can be decreased if a high-speed device is used considerably. The reason is that crossbar chip interconnect delay of a rear riser system begins to become the main causes. The dynamic memory device is not used. In a dynamic memory device, the reason must be regularly refreshed for these and various problems produce it to a rear riser system. When dynamic memory is required, probably the input design section equips the input design section with refresh logic. However, since the realized layout section cannot operate at 100% of layout speed, it is not successful to make the layout section refresh. It is actually desirable to stop activation of the layout section in the case of debugging. That is, the layout sections are some systems, and in order to refresh, they must be dependent on some other components with which the input design section is not equipped. That is, when you need static memory for the layout section, it is unreal to refresh dynamic layout memory. Since a refresh cycle can be disregarded according to static memory, dynamic memory is realizable for layout circles. Thus, layout section memory is materialized using a static device.

[0032] 1.3.1.2 Interconnect RAM and X level crossbar using a single logic chip on the ideal target which uses a logic chip for interconnect with RAM and a crossbar. Under the present circumstances, pins enough like all L-X interconnect pass to connect all RAM signal pins are used. By the practical rear riser system memory module, many pins are needed, so that a single logic chip is difficult to perform. For example, two banks which consist of eight 32 K or 8-bit RAM are assumed to be what is used into the module which has 128 L-X pass. Each RAM bank is equipped with 15 address pins, eight write-in enabling pins, and 64 data pins. Two banks and L-X pass need 302 pins and the pin for a host interface bus. This is twice the number of pins of an usable logic chip. Many logic chips must be used from 1. In the architecture described here, many small logic chips are used and the address, control, and the special function about a data path are given to these chips.

[0033] 1.3.1.2.1 In memory address logic chip drawing 23 , "MA0" and "MA1" show the address and a control logic chip. RAM is divided into a bank. This bank is controlled by each MA chip. MA chip of the maximum number and the same number of separation layout section memory which should be realized with a module is prepared. Only the pass which is required for the each for the address and the control line of the set of L-X pass connected to the crossbar, i.e., a bank, is prepared. MA0 and MA1 use the group of another pass. For example, two independent memory is realizable with two MA chips respectively connected to the one half of RAM. When realizing one large-sized memory, the address and a control circuit network are connected to both MA chip using the group of both L-X pass. Each MA chip controls the address input of all RAM in a bank. An address input is tied up with a single bus. Each MA chip is each, controls the control input to RAM and enables it to write it only in RAM which addressed data. That is, in order to make each MA chip accessible, while connecting with a host interface bus, it has connected with all the

logic chips of this memory module at the common control bus. Drawing 24 shows how MA chip is connected to X level crossbar and a RAM chip further to details. As shown in drawing, MA chip is constituted according to logic and a data path. All the addresses go into MA chip from a crossbar. Usually, (when a bus interface is made into a non-active state), the portion of the address bit equivalent to the number of RAM address bits is passed, and RAM under bank controlled by MA chip is addressed. The decoder logic which controls the write-in enable signal of each RAM by other address bits and write-in enabling [of the layout section] is driven. Logic is constituted according to the configuration needed for this layout section memory. For example, it has the bit width of face as one RAM with the same layout section memory, and when the layout section writes in and it asserts enabling, according to an address bit, one RAM write-in only enabling is asserted. When layout section memory has a chip twice the width of face of one, RAM write-in enabling [of a pair] is asserted. When it desires layout section memory which has many write-in enabling from 1 which is controlling respectively the subset of the data path width of face of memory, some design-specifications lump enabling networks can be used. Each network makes suitable the configuration of the decode logic in MA and MD chip, and operates along above-mentioned Rhine. This is dependent on the availability of the L-X pass connected to MA chip, and the control bus pass connected to MD chip. A host can make this RAM access through a host interface bus by the bus interface logic. When addressing RAM used as this group using a bus, a bus interface switches an address multiplexer ('mux') and addresses RAM to that address. When a host writes in one RAM, a bus interface logic transmits a signal to decoder logic. Decoder logic uses an address bit and asserts RAM write-in enabling [suitable], without driving RAM. Finally, some signals are needed although the data path in MD chip is controlled. Since all the MD chips are not connected to the same L-X pass as MD chip, MD chip does not need to access the address and the control signal from the layout section. A control bus is connected to all MA and MD chips, and it enables it to transmit these signals and a bus interface control signal to MD chip.

[0034] 1.3.1.2.2 A memory data path logic chip MD chip operates a data path according to a bit slice configuration. By intersecting a crossbar and performing a bit slice, the multi-bit bus data path in a rear riser system is interconnected. For every chip, using 1 or 2 bits, the bus intersected X chip and spreads out. The bit slice of the MD chip is carried out, and connection with these buses is made easy. While connecting each MD chip to the same bit in each RAM under all banks, it connects with the subset of X chip. All the same RAM bits can be tied up in MD chip, and flexibility can be given to the configuration of various bit width of face and size layout section memory. By constituting logic and a data path appropriately in MD chip, layout section memory consists of various multiples of RAM width of face. When preparing MD chip of the 'n' individual, and X chip of the 'M' individual, each MD chip is connected using various X chips of M/n. Each data bit is an addition input for DI for a separation I/O configuration and DO pass, or a common I/O bidirection configuration, and either of the addition results because of two L-X pass, i.e., a crossbar addition interconnect configuration. Thus, each MD chip has the L-X pass of a 2*M/n individual at least. In addition to these, additional pass can be prepared. These additional pass can be L-X repeated [of MA]. The number of MD chip, RAM, and RAM bit width of face is chosen, these constraint and capacity constraint are fitted, and this is made to become even number, using effectively the number of the pins in the logic chip used for MD chip. The British Standard static RAM chip has the common I/O structure of having a bidirection data pin (DQ being called), and is used for data in and data out. This has the address input pin (ADDR) and the write-in enabling pin (WE). In this example of implementation, it enables permanently the output enabling pin and the chip-select pin, and an output pin is controlled by write-in enabling. In being required, it performs read-out of RAM, and address data are driven by DQ pin. When it asserts write-in enabling, DQ pin receives data in. At the time of termination of this opinion, data is written in the address location. A specification device needs only the data under setup to the time of termination of write-in enabling, and needs the zero holding time, and is enabling write-in enabling control of a data path by this. When layout section memory needs common I/O, the layout section serves as a tri-state network. This is realized using a crossbar addition configuration. That is, the gate of the drive pin is separately carried out by the enabling, and it is

brought together in the addition OR gate which drives a receiving pin. The interface of the RAMDQ data pin is carried out by the logic and the data path which are constituted in MD chip as shown in drawing 25 . (One bit 'n', i.e., a bit, is illustrated.) The same is said of others. When L chip has the tri-state driver, each MD chip is constituted using the enabling gate which drives the addition gate in X chip as it has the enabling gate which drives the addition gate in X chip (MD 'n' is illustrated). DIZEBURU [logic / a receiving driver] while carrying out the gate of the output to the addition gate of RAM while enabling an output terminal, when DIZEBURU [with a layout section memory input circuit network / writing]. If that is not right, the value of a circuit is transmitted to RAM from the addition gate, and if write-in enabling is asserted, the writing of it will be attained. As mentioned above, it must be cautious of layout section write-in enabling and an output enable signal arising from MA chip (minding a control bus). The bus interface logic is not illustrated. When layout section memory needs separation I/O, this is extracted from common I/O of SRAM as shown in drawing 26 . When output enabling is asserted, data out is always reflecting the data pin state of SRAM. When write-in enabling is asserted, data in is transmitted to DQ pin of SRAM. In the above-mentioned drawing, only one RAM connected to the layout section data bit is illustrated. Occasionally, some RAM is prepared and the number of the locations in layout section memory is the multiple of the magnitude of a single RAM chip in this case. In such a case, MD chip is constituted as shown in drawing 27 . some DQ pins of each RAM are connected to this MD chip. A row address bit, the layout section, and a bus interface control signal are transmitted to MD chip from MA chip through a control bus. In the case of read-out, the low bit of the address chooses any one of the RAMDQ outputs through a multiplexer. The gate of the selected output is carried out by layout section output enabling, and it constitutes layout section memory data out like the above-mentioned example by it. When the layout section asserts the output enabling, data in is transmitted to any one of RAMDQ inputs by enabling a driver. The suitable driver which should be driven is chosen by the decode logic driven with a row address bit and a layout section write-in enable signal. It stops driving write-in enabling [of a RAM chip] with MA chip. Drawing 27 shows the separation I/O configuration. The common I/O configuration is similar to the data in driven by the crossbar addition gate, and the data out in which the gate is carried out by layout section output enabling and write-in enabling, and drawing 25 . It is alike, and an addition gate input is driven so that it may be shown. When a host interface accesses this memory through a host interface bus, the logic constituted with MA chip outputs the control signal for accessing a bus. This signal is transmitted from MA through a control bus. When a bus performs read-out, bus read-out enabling transmits the data chosen from RAM which the multiplexer addressed to the host interface bus data bit corresponding to this MD chip. When a bus writes in, the data from a bus data bit is switched to a driver using other multiplexers. This data is transmitted to DQ pin of RAM chosen by the same process as the usual writing. This explanation must be noticed about MD chip configuration constituted from data path width of face of single layout section memory using the single data bit being shown. When it is based on a layout section memory configuration and you need [and] MD in a module, and the number of RAM chips, many data bits appear in each MD chip from 1 by only bending a data path appropriately. Furthermore, by bending said data path and a control line, using the common MD chip of a lot, much layout memory is realized and some memory is materialized from 1. Only since a certain L-X pass connected to the memory module was connected only to MA chip, and a certain L-X pass is connected only to MD chip, and the network connected to layout section memory using suitable L-X pass is interconnected, a layout section conversion interconnect process is assembled.

[0035] 1.3.1.3 Specify the layout section memory in the input design section using the layout section memory RAM base element corresponding to any one of the available configurations in the layout conversion Original Engineering Consult. file for layout section memory. The layout section conversion method is based on the partial netlist file which the lot defined beforehand. One of these is a thing for [each] the logic chip of a memory module, and it uses the statement about all the logic and data paths that should be constituted in order to carry out a special memory configuration, as shown in the top. Except for the number specification of I/O pins of the module I/O pin used for the file defined beforehand making a layout section memory address,

data, and control connection using interconnect, it is perfect. This method is as follows. : Although there is a special exception to layout section memory as shown below so that the section of layout conversion may describe, : and the layout section reader which uses a general method for layout conversion read the memory base element to specific vector memory to design-data structure. The data for specifying which configuration is used is recorded into the data structure record of memory.
- A conversion stage confirms that a pin is as right as a configuration and the configuration is corresponded available.
- A user tells party SHONA (partitioner) whether the memory module is carried in L chip location of which board top throat. Based on this data, party SHONA chooses the memory module for storage according to a general division algorithm. Alternatively, a user can allocate memory to a specific module by relating this data with the base element in the Original Engineering Consult. file. The layout section reader is equipped with the Original Engineering Consult. file into the base element record of memory.
- Next, an interconnector assigns the network and pin which were connected to memory to specific L-X interconnect pass. An interconnector assigns pass only to the specific pass which has connected the address and a control circuit network to MA chip a condition [constraint that it can assign only allocation and the pass which has connected the data network with MD chip]. These constraint is applied in case it interconnects, when determining the network interconnect capacity of each crossbar chip set and refusing these sets, and when the pass which has not connected MA or MD chip needed cannot be obtained or used.
When writing in the netlist file about each logic chip in a rear riser system, each layout section memory circuit network connection either :1MA or MD - In interconnect procedure The procedure same with having explained, when the determining-whether to connect with pass which base element chooses 2 usual number of logic chip I/O pins was obtained is used. the network of this MA/MD chip that is not assigned to a network besides the obtaining-from numbers-of-passes and number of MA/MD chips-number of logic chip I/O pins 3 former -- since -- Choosing-address, data, or control connection defined beforehand 4 statement A netlist is carried out by specifying using for connecting with the layout section memory connection which defined beforehand this number of logic chip I/O pins in addition to the netlist file of this logic chip.
- While processing a netlist file to a configuration bit pattern using a netlist conversion tool, load to the logic chip as a netlist file of L chip and X chip.
[0036] 1.3.1.4 The concrete memory module engineering drawing 28 is drawing which is used in a suitable example and in which showing the layout section of a memory module. It must be cautious of acting as the architect of this according to the configuration based on the above-mentioned explanation shown in drawing 23 . XC3090 It constitutes so that a plug may be inserted in L chip socket replaced with a LCA logic chip and it may connect with it. Thus, 128 L-X pass, i.e., four pass which has led to 32 X chips respectively, is prepared. 32 K or 8-bit static RAM chip which has common I/O is used during each eight banks [two] of RAM. Each bank is MA chip of itself, and XC2018. It has LCA. Each MA chip controls the RAM using eight address pass and eight write-in enabling. While connecting each MA chip to the control bus which all MA and MD chips in a module are sharing, it connects with a host interface bus. The remaining pins are connected with the crossbar. 28 L-X pass connected to a respectively different X chip is prepared. The MA chip 0 uses the pass of a lot, and pass 0, and MA1 uses pass 1. The separate address and the control circuit network to two independent layout RAM are given by this. Pass fewer than 32 perfect L-X pass is connected. This is only exactly because the number of the pins of XC2018 is restricted. It must be cautious of the ability of the pass component in an interconnect L-X pass table corresponding to the missing pass in this module not to be used during layout conversion. For this reason, a network cannot be interconnected through a pass component. In eight MD chips, it is XC2018 altogether. LCA is used. When preparing 32 X chips, according to the above-mentioned method, each ND chip has connected 32 / 8= 4 different X chips. Each chip has the 2*M/n=8 piece pass used for a layout section memory data bit. Two of pieces [them] have led to X chips each. Two additional pass connected to X chips each is

prepared, and it enables it to use a module as 128 bit-vector memory, as shown below. The host interface bus realized in a suitable example is called R bus. R bus connects all L chip positions using an additional pin. The section of a host interface explains this. Five different layout section memory configurations can be used in this module. In the following charts and drawing 28 , "pass 0" shows the L-X pass of the lot connected from X chips each, and "pass 1" shows other lots.
- One memory of 8-bit 512K : 16 data through the 19 (it is made the duplex so that it can connect with both MA0 and MA1) addresses through the L-X pass 0 and 1 and two control (WE, OE), and the L-X pass 2 and 3 (DI/DO, or a driver/receiver). Each MD chip has one data bit connected to 16 RAM.
- One memory of 16 bit 256K : 32 data through the 18 addresses through the L-X pass 0 and 1 and two control, and the L-X pass 2 and 3. Each MD chip is equipped with two data bits respectively connected to eight RAM.
- One memory of 32-bit 128K : 64 data through the 17 addresses through the L-X pass 0 and 1 and two control, and the L-X pass 2 and 3. Each MD chip has four data bits respectively connected to four RAM.
- Two memory of 8 bit 256K : it has the 18 addresses through L-X pass, and two control respectively. Pass 0 is a thing for one memory (MA0), and pass 1 is a thing for the memory (MA1) of another side. Each has 16 data through pass 2 and 3. Each MD chip has one data bit for each memory connected to eight RAM.
- Two memory of 16-bit 128K : each has the 17 addresses and two control through L-X pass. Pass 0 is a thing for one memory, and pass 1 is a thing for the memory of another side. Respectively, it has 32 data through pass 2 and 3. Each MD chip has two data bits for each memory connected to four RAM. The control bus consists of 12 pass generally connected to all MA and MD chips. Twelve pass needs to hold the maximum control configuration. This configuration is three address bits. That is, it adds bus write-in enabling and bus read-out enabling to design-specifications lump enabling and the layout section output enable signal for [each] two 256 K or 8-bit layout section memory.

[0037] 1.3.2 It is dependent on stimulus signal transmission of a host computer, and prehension of the reply signal to the layout section, and the reply signal from the layout section to use many stimuli and response rear riser systems. Vector memory is used when performing this in batch format (i.e., when transmitting and collecting most signals at once). When one signal performs this at once, SUTIMYURETA and a sampler are used.

[0038] 1.3.2.1 the vector memory for giving a stimulus -- it is sometimes necessary to supply the stream of a continuous and repetitive stimulus to the network of the lot in the layout section realized for high speed iteration application of a test vector like simulation application. This is performed making the network of the layout section realized carry out the interface of the memory, writing a stimulus vector in memory from a host computer, and by sending memory to read-out 1 time thru/or several times, and sending a stimulus to the layout section further one by one. Since it is necessary to read a continuous and linear memory location, an address stream is prepared by the binary counter. Drawing 29 shows the means for attaining such stimulus vector memory. Regular clock signal ECLK controls a process. A low is generated for ECLK as yes for whenever [of period-izing, i.e., each stimulus vector, / every]. A binary counter offers address sequence. If ECLK becomes a high, a counter will be counted up to the address of the following stimulus vector. The address of the following stimulus vector is read by between [RAM] the periods of ECLK. If ECLK becomes a high next, the value of the stimulus vector read exactly will serve as a clock of a D flip-flop. The output signal of a flip-flop drives the circuit stimulated with the value of a stimulus vector. A flip-flop gives clean tolan JISSHON required between vectors. The reason is because it may change between the read-out cycle, before stabilizing a RAM output in a right value. This process is repeated and is given to the layout section by which a series of stimulus vectors are realized. This structure is repeated and many networks are provided with a stimulus. Since the interface to the host computer used for writing a stimulus vector in RAM is easy, although it is not illustrated, the drawing quoted below shows it to details further.

[0039] 1.3.2.2 Catch the vector from the stream of a continuous sample, i.e., the network of a

lot, in the 1 mode which catches the response from the layout section realized like the vector memory for response prehension. At this time, a logic analyzer performs prehension from an actual hardware device. This makes the network of the layout section realized carry out the interface of the memory, and when the layout section realized operates one by one, it is performed by returning the vector from a network to writing and returning the response vector caught further to a host computer at memory for analysis. Since it is necessary to read a series of continuous and linear memory locations, an address stream is prepared by the binary counter like the above. Drawing 30 shows a means to develop such response vector memory. Like a stimulus mechanism, clock signal ECLK controls a process. The synchronization of ECLK is taken for whenever [of each response vector / every]. A binary counter offers address sequence. If ECLK becomes a high, a counter will be counted up to the address of the following vector. If ECLK becomes a low, the value of a response vector will be transmitted to a RAMDQ data pin with a tri-state driver, and it will enable RAM for writing. If ECLK becomes a high again, this value will be written in a RAM location, and are DIZEBURU RAM write-in enabling and tri-state driver enabling, and a counter goes to the address of the following vector. This process records a series of response vectors of the layout section repeated and realized. This structure is repeated and a stimulus is supplied to many networks. Although the interface to the host computer used in order to write a stimulus vector in RAM is not illustrated since it is easy, it is further explained to details in the drawing quoted below. Generally the layout section realized is stimulated and these responses are generated. When a stimulus arises from stimulus vector memory, both vector memory uses the same ECLK signal. An ECLK signal needs to be a high sufficiently long [although data is read and a stimulus D flip-flop is set up] while the new address is read in a counter and it addresses RAM. Moreover, an ECLK signal must be a low sufficiently long [although the layout section by which a stimulus is realized is affected, and all responses to this effect are stabilized and these responses are written in RAM]. When a stimulus arises from either, in order to sample a response network correctly, the ECLK signal of response vector memory needs to synchronize with the layout section realized.

[0040] 1.3.2.3 The function of the stimulus and response vector memory which were defined as mentioned above about the stimulus and the response vector memory system is combinable as shown by vector memory drawing 31 for a stimulus and a response. Even if, even if a RAM bit is the same RAM device, it can be freely assigned to either a stimulus or a response. The reason is that a stimulus read-out function arises when ECLK is a high, and a response write-in function follows this when ECLK is a low. One bit can be used for both a stimulus and read-out by considering a tri-state response driver both as a stimulus D-flip-flop input, and connecting with the same RAMDQ data pin. I hear that the material-difference point of simple stimulus vector memory and combination stimulus / response vector memory can read a stimulus vector from RAM only once, and there is. The reason is because each memory location is written in at the time of the low of the half period of ECLK, even when a RAM bit is used only for a stimulus. This can be avoided, only when the bits of a RAM chip are used [no] for a stimulus and ECLK asserts write-in enabling. The front drawing is illustrating what realized vector memory by the general method. Furthermore, a dotted line shows how a vector memory logic function is realizable with constituting a logic chip ("MA chip" and "MD 'n'"). These logic chip is appropriately connected to a RAM chip and rear riser interconnect (X chip). The conversion which returns the stimulus from software to an electric mold again is explained to details as vector memory in the U.S. Pat. No. 4,744,084 specification. These contents are used here for reference.

[0041] 1.3.2.4 Explain the vector MEMORIRIARAIZA fault simulation system for fault simulation in the section about this. By fault simulation, a response is not caught by vector memory, instead is compared with the response of a good predetermined circuit by fault response vector memory. Fault response vector memory is the same as the simple stimulus vector memory shown in the following points in the top. That is, an output is measured with the value of a network by the XOR gate instead of driving a network using the output of the flip-flop of MD chip. It connects with the set flip-flop with which ECLK takes a synchronization for an XOR gate, and a flip-flop is set when the XOR gate which shows the difference of a network and memory is

a high. A host can read this set flip-flop through a host interface, and can investigate whether the difference is detected or not.
[0042] 1.3.2.5 The connection method of the vector memory to the layout section by which interconnect implementation of the vector memory in the layout section realized is carried out can consider many methods. A rear riser system can be designed using the vector memory which direct continuation was carried out to one or more logic chips, and/or was connected to all interconnect all [either or]. For example, while being able to attach vector memory in a logic board using L chip and X chip, it is connectable with a board at separated X-Y pass. While attaching vector memory in Y chipboard of a Y level crossbar, it is also connectable with X-Y and Y-Z pass. There is also a technique of connecting vector memory to the L-X pass which acts on installation and L chip location at L chip location instead of a logic chip. In this case, these L-X pass is connected only between vector memory and X chip. By constituting X chip, connection with the network of the layout section realized is made, and vector memory is connected to a network. Under the present circumstances, the network is connected through X level interconnect. A logic chip is transposed to a vector memory module by the modular method, it is a required number or a rear riser system can consist of necessities using a small number of vector memory. Since rear riser layout section memory was replaced with one or more logic chips under L chip location and is attached, a common hardware memory module can be used as a layout section memory module or a vector memory module using this technique. While constituting a logic chip in a memory module, a function is chosen by interconnecting a rear riser system appropriately. This is vector memory architecture used in the suitable example.

[0043] 1.3.2.6 Use a common memory module in a special vector memory layout section suitable example for both layout section memory and vector memory application. The general architecture and layout are explained in the section of layout section memory, and are not explained here. It is as the details of how to use a module as vector memory being shown below. The following two drawings show that MA for combination stimulus / response vector memory and the logic same in MD chip as the above are constituted using completeness read-out / write-in access from a host interface. the case where a host computer is a non-active state -- all actuation -- the above -- the same technique as the easy example having shown is followed. In drawing 32 , the ECLK signal which a host outputs through a host interface is interconnected for MA chip through interconnect. An ECLK signal takes the synchronization of the address counter which consists of each MA chip. Since one or more MA chips which are controlling RAM of a lot are respectively prepared into the module, each MA chip has the copy of a vector address counter. Since all counters have obtained the same control (reset signal from ECLK and a bus interface), the each always transmits the same address as other counters. Usually, the address is sent from a counter output and RAM is addressed (when a bus interface is a non-active state). In the case of a low state (write-in response phase), as for decoder logic, ECLK asserts all RAM write-in enabling like the above-mentioned example. ECLK is transmitted also to a control bus and drives the logic of MD chip. MD logic processes a stimulus and response vector value itself (drawing 33). Usually, when ECLK is a high state, RAM synchronizes RAM and a flip-flop, if read-out and ECLK will be in a low state about a stimulus vector value (when a bus interface is a non-active state). A flip-flop is for giving a stimulus like the above to each network (one of them is illustrated). Therefore, a stimulus is transmitted to a network through an interconnect X chip. When ECLK is a low state, all tri-state enabling (e0, e1, ... en) are asserted, and the response value outputted from a network through interconnect (two pieces are illustrated) is transmitted to a RAMDQ data pin through a multiplexer. When a host computer accesses this memory through a host interface bus (especially R bus of a suitable example), the bus interface logic constituted in each MA chip will be in an active state. As for this, a switch and a bus address RAM for an address multiplexer (mux). When it is for a bus cycle to write in RAM, decoder logic outputs a suitable write-in enable signal while decoding in which RAM it should write using the address bit. The address bit, read-out, and the write-in control signal which are needed for choosing RAM are also transmitted to MD chip through a control bus. In MD chip, when a bus performs a read-out cycle, decoder logic all tri-state RAMDQ pin drivers,

DQ data output of RAM addressed through the read-out multiplexer using the address bit is chosen, and a bus read-out enable signal transmits a data value to the data line of the host interface bus for this bit further. In a bus write cycle, a decode logic enables the tri-state RAMDQ driver for addressed RAM, and transmits data to a RAM input while it chooses not the network that gives a response but the data value produced from the data line of a host interface bus using a write-in multiplexer.
[0044] 1.3.2.7 in order to explain that layout conversion and the specification network of vector memory should be connected to vector memory, a user adds the special feature about an input design to a network, and it is a thing for a stimulus of specific vector memory and connection -- or explain whether it is a thing for a response. Based on the partial netlist file of a lot predetermined in the layout section conversion method, one of these is a thing for the logic chip of each module, and the statement about a vector memory stimulus and response connection, a vector memory data path and control logic, and a bus interface logic is used like the above. An ERCGA netlist conversion tool does not constitute the logic for a base element in the netlist file which is not usually connected like the output terminal which is not connected to the input terminal which is not connected to the output terminal or I/O pin of arbitration and the input terminal of arbitration, or the I/O pin, and a network, and interconnect from this method. Logic is prepared for the stimulus connection to each vector memory bit, and response connection. Only one which is supplied to a netlist of interconnect is actually constituted, and another side is not constituted. The reason is that it does not usually connect it to a netlist. The file defined beforehand is perfect except for the specification of the number of I/O pins of a module I/O pin used for connecting vector memory stimulus connection and vector memory response connection using interconnect. It determines "To use how many I/O pins for the number of the stimuli and response connection in each file into the logic chip of a file, or to prepare what logic in each chip, and prepare what logic in a module further as a whole, or to be alike." The method is as follows. : Although it have an exception with the following special vector memory as explain in the section of layout section conversion , : and the layout section reader which use a general method for layout section conversion build into the layout section data structure one or more vector memory base elements by which the property information from an input design file be connected to the network instead of read-out and a bus interface logic , in order to identify the network prepared for vector memory connection . A layout section reader makes the ECLK network connected to the host interface clock generation machine and the vector memory base element.
- one on the board on which a user attaches a memory module with party SHONA of chip assignment -- I hear that it carries out and be. Based on this data, party SHONA divides a vector memory base element into a memory module by the usual method.
- An interconnector processes the same vector memory base element as other logic chip base elements, and determines the L-X pass which has connected these using other base elements in a network.
- :1 to which the netlist of each vector memory circuit network connection is carried out by the following when writing in the netlist file of each logic chip in a rear riser system -- determine which logic chip connects the pass which the base element chose in interconnect procedure.
2) Obtain a logic chip I/O pin number from a pass number and a logic chip number using the procedure same with having explained, when obtaining the usual logic chip I/O pin number.
3) Choose the stimulus or response vector memory connection defined beforehand from the network about the logic chip which is not assigned to a network besides the former.
4) Add a statement to the netlist file of this logic chip, and specify using in order to connect this logic chip I/O pin number to the vector memory connection defined beforehand.
- Layout conversion system also sends out a correspondence table file, relates the name of a network with vector memory and the vector memory bit position, and carries out working use.
- An ERCGA netlist conversion tool constitutes only the logic of the vector memory stimulus and response input terminal which are used, and interconnect.
[0045] 1.3.2.8 SUTIMYURETASUTIMYURETA considers as a single storage bit, control it by the host computer, and it drives the network of the layout section. SUTIMYURETA is used for a host

supplying an input signal to the layout section. Two kinds of SUTIMYURETA, i.e., a random access type and an edge detection type, is prepared. Actual random access SUTIMYURETA is a flip-flop, and the output signal drives the layout section network where a host loads data if needed through a host interface bus. Random access SUTIMYURETA is used for stimulating the network to which a value can always be changed in response to the network where others were stimulated, without changing actuation of the layout section. The data input to a register occurs as an example of such a network. When each SUTIMYURETA has the only bus address and a host writes data in this address, a bus interface logic takes the synchronization of the clocked into of a SUTIMYURETA flip-flop while giving data to D input (drawing 34). Edge detection type SUTIMYURETA is used for stimulating the network which must change while synchronizing with other networks for correcting actuation of the layout section, for example, the clocked into to a register. The 2nd flip-flop is arranged between random access SUTIMYURETA and a layout section network. a group which must take a synchronization -- such all SUTIMYURETA is connected to a common clock. In order to input the circuit value of a new lot, no matter a host may be what order even if, he loads a new value to the 1st flip-flop of each SUTIMYURETA through a host interface bus like the above. When a new value needs to be altogether supplied to the layout section, a host period-izes in common 'a synchronous clock', loads all values to the 2nd flip-flop at once, does in this way, and drives all networks to coincidence (drawing 35).
[0046] 1.3.2.9 A sampler sampler is a single storage bit, is controlled by the host computer and receives the network of the layout section. A sampler is used by the host and catches the output signal from the layout section. The easiest form of a sampler is the flip-flop which can receive a layout section network with D input terminal, and can take a synchronization, and a host can read if needed through a host interface bus and a bus interface logic. Usually, many samplers are connected in common 'a sample clock'. Sampler data output has the only bus address like the 'sample clock' output. a host -- a clock -- a period ---izing -- a group -- the data values sampled ejection and after that in the sample are read one by one (drawing 36). In order to reduce the number of host I/O needed, the 2nd flip-flop is added additionally and a change detection sampler is constituted. The 2nd flip-flop is connected to the same clock as a sampling flip-flop, and the input terminal is connected to the output terminal of a sampler. As a result, the 2nd flip-flop holds the value which the sampler had before the newest clock period. An XOR gate compares the flip-flop output of two pieces. An XOR gate outputs the value of a high state, when two flip-flops are different for the sampled value change. a group -- a host adds the total XOR output signal from a sampler by the OR gate in which read-out is possible. As mentioned above, after sampling a network by period-izing a 'sample clock', a host checks the 'change' value of this OR gate to the 1st first, and investigates which value in a group changed. When not changing, it is not necessary to read any values of these samplers (drawing 37).
[0047] 1.3.2.10 Realize SUTIMYURETA, layout conversion of a sampler, a specification sampler and a SUTIMYURETA flip-flop, a logic gate, and a bus interface logic in a RIARAI SYSTEM logic chip. In order to explain that a network should be connected to a sampler or SUTIMYURETA, a user gives the special property about an input design to a network, and identifies the specific type of SUTIMYURETA or a sampler, and a group's identity. Constitute SUTIMYURETA and a sampler, and in order to connect with the remaining portions and bus interfaces of the layout section, these As explained in the section of :layout conversion which is as being shown below, the general method using a layout conversion software system Although there is a special exception about the following SUTIMYURETA and samplers, : and the layout section reader which uses a general method for layout section conversion The network in which property information was prepared from the input design file for read-out, SUTIMYURETA, and/or a sampler is identified, and the base element of the SUTIMYURETA and the sampler which were connected to the network instead of a bus interface logic is built into design-data structure.
- System party SHONA has the data base about how many equivalence gates each of such a base element has in the logic chip. System party SHONA also has the equivalence gate characteristic of a bus interface logic. Based on this data, system party SHONA assigns SUTIMYURETA and a sampler to a logic chip according to that usual division algorithm. Under the present circumstances, the additional conditions that system party SHONA makes the limit

of logic capacity small with the size of a bus interface logic are imposed, and each of a logic chip which has one or more SUTIMYURETA and/or a sampler explains that it must have a bus interface logic block.
- An interconnector processes SUTIMYURETA and a sampler base element as well as other base elements.
- When writing in the netlist file of each logic chip in a rear riser system, carry out the netlist of each sampler or the SUTIMYURETA base element using the following procedure.
1) Transmit the fundamental statement of the gate which constitutes a sampler or SUTIMYURETA, and/or a flip-flop to the netlist file of the logic chip for statement division. The name of the additional network covering the network sampled or stimulated according to the method same with having explained about an interconnect base element is obtained from the name of a network sampled or stimulated.
2) When this is the 1st SUTIMYURETA and the sampler by which a netlist is carried out to this special logic chip file, supply the base element and network of a bus interface which constitute a bus interface using the netlist file segment defined beforehand to a logic chip. The standard name defined by said file segment is given to bus interface interconnect used once for every interface. The name of the caught network is given to what is connected to SUTIMYURETA or sampler logic. This name is adjusted with the name used when outputting a base element in step 1.
Although it is easy, SUTIMYURETA and a sampler are realized using the method which is not common only in the logic chip of a memory module or the device module specified by a user. This assumes not constituting the logic for a base element in the netlist file which is not usually connected like the output terminal connected to the input terminal by which the ERCGA netlist conversion tool is connected to no output terminal or I/O pin and no input terminal, or I/O pin, and a network, and interconnect. This is based on the partial netlist file by which the lot was defined beforehand. One of the file of this is a thing for the logic chip of each module. Under the present circumstances, the following statements are used.
1) SUTIMYURETA of much edge detection types connected in common 'a synchronous clock' altogether.
2) Many change detection samplers altogether connected to the same 'sample clock'.
3) The bus interface logic for [all] the above.
The file defined beforehand is perfect except for the specification of the I/O pin number of the module I/O pin used for connecting a sampler and SUTIMYURETA using interconnect. A common signal like a synchronization and a sample clock is distributed between logic chips using a control bus. How many I/O pins can use into the logic chip of a file, or each chip has what logic, and can carry out the thing of, or the module as the whole determines SUTIMYURETA in each file, and the number of samplers. The method uses for layout conversion a general method which was explained in the section of :layout conversion which is as being shown below. Under the present circumstances, there are the following exceptions about SUTIMYURETA and a sampler. : - layout section reader builds into the data structure of the layout section SUTIMYURETA and the sampler base element by which the property information from an input design file was connected to the network instead of read-out and a bus interface logic, in order to identify the network prepared for SUTIMYURETA and a sampler.
- I hear that a user specifies one on the board furnished with a memory module and the device module specified by a user of L chips as party SHONA, and be. SUTIMYURETA and a sampler base element are divided to the remainder of such a module until it reaches a number of limits with which party SHONA can use memory and a USD base element for a module in each module unit allocation and after that according to that usual division algorithm based on this data first.
- An interconnector processes other the same SUTIMYURETA and samplers as a logic chip base element, and determines the L-X pass which has connected these in the network using other base elements.
- :1 to which the netlist of each sampler or the SUTIMYURETA base element is carried out by the following when writing in the netlist file of each logic chip in a rear riser system -- determine which logic chip connects the pass which the base element chose in interconnect procedure.

2) Obtain a logic chip I/O pin number from a pass number and a logic chip number using the procedure same with having explained, when obtaining the usual logic chip I/O pin number.
3) Choose SUTIMYURETA/sampler defined beforehand from the network about the logic chip which is not assigned to a network besides the former.
4) Add a statement to the netlist file of this logic chip, and specify using in order to connect this logic chip I/O pin number to the sampler/SUTIMYURETA defined beforehand.
- An ERCGA netlist conversion tool constitutes the logic for SUTIMYURETA to be used, a sampler, and a related bus interface logic, and interconnect.
In both methods, layout section conversion system communicates the address with a host interface bus while relating a network name to specific SUTIMYURETA and a specific sampler, in order to also output a correspondence table file and to use it working.
[0048] 1.3.3 In order to realize the input design section in the hardware which actually operates with the gestalt of the logic and the interconnect chip by which the user assignment device configuration was carried out, it is practical to connect other actual hardware devices to a rear riser system, and it is desirable. A microprocessor or other VLSIs The device of arbitration equipped with digital input/outputs, such as IC chip, a digital/analog converter, a display device, an input keyboard and a switch, a storage device, and computer input / output bus, can be formed. These can be made into a part of digital system which constitutes a part of [like a circuit board or a large-scale scale configuration element] layout section realized. These devices show a part of input design section which cannot be embodied in the logic gate of a rear riser system, a flip-flop, and memory and which should be realized. Since this is based on a physical reason like a display and is insufficient of resources of a rear riser system like a large-scale storage device, it is because logic-description cannot be used like a standard microprocessor. instead, these devices are already constituted -- having -- the right -- things are proved -- a half -- usually -- it can also become the device which does not desire a thing like a gate array chip which a user realizes using a rear riser system resource. Since a rear riser system resource does not need to be used for the reason in order to realize this, it is because a user wants to examine the exact actuation using the device of a portion with which the layout section is realized. Although these devices are not some rear riser systems, since they are what is specified by the user according to the necessity for layout of a user, they call these devices a "user assignment device" (USD). Various USD(s) which are useful to forming a standard means which is used for a user connecting such a device to rear riser system hardware in a rear riser system are prepared. This means is a device module (USDM) specified by a user.
[0049] 1.3.3.1 the device module specified by [specified by a user] a device module user -- :1 -- it has a means to connect the hardware device specified by a user physically.
2) Between USD, rear riser system logic, and/or interconnect chips is connected. In order that USD may play the role of a logic chip and the similar layout section, it is the same method as a logic chip, and it is convenient to interconnect USDM.
2) Usually, prepare the function which assigns a USD pin freely to an interconnect pin so that the logic chip attached in L chip location may carry out.
Since the device module specified by a user needs to be equipped with the function similar to the memory module having in the RAM chip, the architecture of USDM is similar with the architecture of a memory module. Drawing 38 shows USDM architecture. A device is attached in such other fields connected to the USDM printed circuit board, i.e., USDM, through the cable by the method of being common in the device installation field specified by [which is the field of the movable daughtercard (daughtercard) attached with the plug] a user or a microprocessor, and an emulator (emulator) meter. The block of a terminal is equipped with the means for making electric connection between a device I/O pin and a USDM logic chip through connector terminal thin **, the printed circuit board pad of a lot, or such other means. The power supply of a device is also equipped with the block of a terminal. As long as a physical capacity of a terminal block pin allows, one or more devices can be attached. Instead, a device is also remotely connectable through a cable and repeating installation by the general method. Each of MA and MD logic chip is equipped with the I/O pin connected to the terminal block, and the I/O pin connected to interconnect. It connects with interconnect by the method same with having explained these

chips in the memory module address and a data path logic chip. Additionally, as shown by a diagram, these chips are connected also to a host interface bus and/or a common control bus for the purpose same with using a chip for a memory module. The USD address and a data bus are connected to MD chip so that a bus data bit may be distributed to MD chip and may generally be distributed to interconnect by this. MA chip is used for a USD control line and an addition target at a USD address line. The drawing shows three hypothetical user devices connected in order to explain a possibility. USD0 has the data and the address bus which were connected through MD chip, and the control lines A, B, and C connected through MA0. USD1 has three data buses connected to MD chip, and the addresses through MA chip and control connection. USD2 uses MA1 for addressing, and MD chip for data. In the specific case of arbitration, the user of a rear riser system can connect these USD(s) using a suitable method for these layout and use. As said section was shown, in a memory module MD chip, bidirection USD connection is interconnected using the method same with interconnecting a bidirection RAMDQ pin. Differences are the requirements that it is necessary to carry out the network of the input design section like output enabling control, and to specify it. This network is connected to interconnect logic by the same method as "layout output enabling", and the bidirection driver of MD chip is controlled. [which is shown when the numbers of memory modules are 25 and 26] Usually, when the suitable output enabling control circuit network is not formed into the input design section, a user needs to prepare this.

[0050] 1.3.3.2 In the suitable example shown in USDM drawing 39 of a suitable example, USDM is the same as that of a rear riser memory module about the field for attaching USD instead of a RAM chip. Each of eight MD chips interconnects the USD pin to 16 pieces, and each of two MA chips interconnects the USD pin to 23 pieces. Drawing Two actually attached VLSI devices, Namely, the 32 bit microprocessor of Motorola MC 68020 ("MC68020 32 Bit Microprocessor User's Manual", Motorola, Inc., Phoenix, 1984), The Motorola MC68881 floating-point coprocessor ("MC68881 Floating Point Coprocessor User's Manual", Motorola, Inc, Phoenix, 1985) is shown. These devices are the examples which were excellent in USD. The reason is that these devices generally cannot be used for layout of a digital system, and it cannot make these logical-circuit network expressions available [a user]. These devices have the following input-and-output pins, and are as these details being shown below.

MC68020 data : D31-D0 and bidirection output enabling condition:R/W show "writing", and when DBEN is truth, D31-D0 transmits an output signal, and when that is not right, it receives an input signal.

Address :A31-A0, an output central input terminal : [CLK, DSACK0, DSACK1, AVEC,] CDIS and IPL0- IPL2, BR, BGACK, RESET, HALT, BERR central output terminal:R/W, IPEND, BG and DS, DBEN, and AS, RMC, OCS, ECS, SIZ0 and SIZ1 -- FC0-FC2MC6888A data : D31-D0 and bidirection output enabling condition:R/W show "read-out", and when DSACK0 and/or DSACK1 are truth, D31-D0 transmits an output signal, and when that is not right, it receives an input signal.

Address : A4-A0, input central input terminal : CLK, SIZE, RESET and AS, R/W, DS, CS central output terminal : DSACK0 and DSACK1 data bus and an address bus are interconnected using MD chip. As explained in the section of a memory data path, a crossbar is crossed for a bus data bit and it slices, and interconnect is made easy as shown in drawing. A control signal interconnects with MA chip. An output enabling control signal is generated by the special logic connected to the control signal, as mentioned above. A user prepares this logic in the input design section, and is realized in L chip using the remaining portion of the layout section. The group of the L-X pass with which each MD chips differ is connected, and since output enabling control is usually common about all buses, layout section conversion system constitutes MA and MD chip so that it may connect with MD and MA chip with the need for connection of a network using a USDM control bus, while connecting these networks to any one in MA chip.

[0051] 1.3.3.3 Form the conversion USD of the layout section for the device specified by a user in input design circles using a special base element. USD transmits the property data in which the USD specfile which a user creates is shown. This file lists the I/O pin of USD while showing whether USDM which has this device for which L chip location is attached. Under the present

circumstances, the pin name currently used into the USD base element of the input design section is used. USD lists the USDM logic chip which has connected the pin and the number of pins, and that a pin is an input, an output, or bidirection to each pin. When a pin is bidirection, the name of the output enabling control circuit network in the input design section is also listed. A layout section conversion software system outputs the netlist file which connects USD to the remaining portion of the layout section while constituting USD. Although a general method is used, : and the layout section reader which has an exception over the following USD(s) in this read a USD base element into a layout section data structure. A layout section reader memorizes the information relevant to the base element storage for next use while using a file property, in order to perform read-out in USD specfile. Base element storage is supplied to the special pin connected to a respectively different output enabling control circuit network.
- Confirm that a pin is right and the conversion stage corresponds with the configuration available [a configuration].
- System party SHONA arranges USD for L chip location specified by USD specfile.
- An interconnector assigns the network connected to the USD pin to specific L-X interconnect pass. In case an interconnector performs this, it is subject [to constraint that it can assign only the pass which connects the network connected to the USD pin to MA or MD chip specified by USD specfile, and an enabling control circuit network pin can be assigned only to the pass linked to MA chip].
- in order to transmit a netlist file to USDM -- : -- : by which each output enabling control circuit network which is controlling USD of this USDM transmits :base element to the netlist file of MA chip of this network -- that reason is that the input buffer which has received the L-X pass used for this network drives the input of an output buffer, and drives the control bus line assigned to this network by this. : which sends out a base element to the netlist file of the logic chip of this pin when each network connected to USD of this USDM drives :USD input pin -- that reason is that the input buffer from the reception pass used for this network drives the input terminal of the output buffer which drives the terminal block pin used for this USD pin. : which transmits a base element to the netlist file of the logic chip of this pin when receiving a USD output pin -- that reason receives the output of an input buffer whose output buffer connected to the drive pass used for this network has received the terminal block pin used for this USD pin. : which transmits a base element to the netlist file of the logic chip of this pin when having connected with a USD bidirection pin -- that reason The data input terminal of a tri-state output buffer with which the input buffer from the reception pass used for this network is driving the terminal block pin used for this USD pin is driven. An output buffer connected to the drive pass used for this network The output of 2 input AND gate where the input buffer which receives the terminal block pin used for this USD pin drives one input terminal is received. It is because the input buffer connected from the control bus line assigned to the output enabling control circuit network of this pin drives the enabling input terminal of a tri-state output buffer, and other input terminals of the AND gate.
[0052] 1.4 As explained in the section of configuration logic and interconnect chip technology, the configuration bit pattern of each chip is outputted by the ERCGA netlist conversion tool. The last stage of rear riser layout conversion system catches the data sent to the single binary configuration file of the layout section from the configuration file which all chips generate. Data is eternally recorded into a host computer by this. Before using a rear riser system respectively, read-out is minded for data from a configuration file, a host interface is minded for this data, it transmits to rear riser hardware, and the logic chip and interconnect chip of the layout section to be used are constituted by loading to a chip further. Configuration connection is prepared between a host interface, and all the logic chips in a system and an interconnect chip. A configuration of a chip can operate all logic functions and the sum total of interconnect in the layout section in accordance with what is specified by the input design section. In a suitable example, LCA made from Xilinx is used as a logic chip and a crossbar chip. LCA is constituted by loading 1 bit of binary configuration bit patterns to the serial shift register of LCA configuration memory at each 1 time. While supplying each bit to a configuration data input terminal (DIN), a configuration clock (CCLK) is period-ized once and it loads. The special configuration connection

between each LCA and a host interface is not prepared. The reason is that a system must be equipped with the logic chip and crossbar chip to 3520 pieces in all. Instead, the configuration bus which has a multi-bit data path and a configuration clock is formed, and this is connected to all the boards that have LCA. In order to constitute, for every loop, the chip of the number of bits and the same number in a data path is used, and grouping of a logic chip and the crossbar chip is carried out. All the chips in 1 group are constituted in juxtaposition. Each LCA in a group has the configuration data input terminal connected to the bit from which a bus data path differs as shown in drawing 40 . Configuration control logic block in each group is connected to a host interface bus, a bus arrangement clock, and the clock input terminal of all LCA in a group. The group of LCA to which these control logic block is enabled alternatively and a host interface bus means it with an instruction of a host through a host interface bus enables it to receive a clock signal, and is considering as such a configuration. This is a procedure carried out to a host computer constituting a rear riser system. : to which control and data transmission are altogether performed through a host interface -- since all logic chips and crossbar chips are constituted -- :each configuration group -- : -- this group's control logic block directs to send a configuration clock to a chip. Between the configuration bit in one LCA, and the periods of the same number: Load 1 configuration bit of each chip in this group to a bus data path. A bus arrangement clock is period-ized once. To the following period This group's control logic specifies that it does not transmit a configuration clock any longer. To the next group

[0053] 1.5 A rear riser system operates as a peripheral device under control of a host interface host computer. A host computer constitutes the logic chip and interconnect chip of a rear riser system according to the layout section using the configuration bit pattern memorized in the configuration file of the layout section. A host computer controls actuation of the continuous layout section by controlling the external reset and clock signal. Therefore, a host computer acts the contents of a vector and layout section memory commutatively with the layout section read-out and by writing in while controlling SUTIMYURETA, a sampler, and vector memory. A host computer performs these [all] through a rear riser system host interface. A host computer controls the host interface and configuration bus of a rear riser system.

[0054] 1.5.1 Constitute a host interface architecture RIARAI SYSTEM host interface along Rhine which is used commonly completely (drawing 41). The rear riser system is equipped with the host interface bus controller, the configuration bus controller, the clock generation machine, and the reset controller. These each is explained below. While constituting an interface on the board of a rear riser hardware chassis, it connects with the I/O bus of a host computer through a cable and an interface card. The control function of a host interface is created according to the demand of a specific computer in either the memory address space of a host computer, or an input-output bus space.

[0055] 1.5.2 Connect a host interface bus host interface bus to some of logic chips of normal in a rear riser system and memory module logic chips, or all I/O pins. The host interface bus is equipped with the address space which assigns a rear riser system control and a data access facility. A host is the optimal bus master and sends out the read-out command and the write-in command which were addressed to a bus through a host interface bus controller. A host transmits data between a rear riser system function and a host. Host interface control logic block is programmed for the Maine logic chip and a memory module logic chip, and it enables it to control a rear riser system function through a bus. As a special example of the function controlled by this bus, there are a sampler, SUTIMYURETA, vector memory address assignment, operation, a host data access, and a layout section memory host data access. Since these control blocks are altogether programmed by the logic chip, programming of a logic chip can prescribe all of the special function and location in a bus address space, and they can change them the predetermined layout section of arbitration, or if needed for a mode of operation. The specific layout section of a host interface bus is dependent on the data-access speed at the time of materializing a specific rear riser system, and the availability of a hardware pin. In the suitable example, 11 pin host interface bus called R bus is connected to the exclusive I/O pin of all logic chips. The hardware of a suitable example is equipped with eight bidirection Rhine used for data and the address, a clock, and two control lines. R bus has 32 bit-address space and 8

bit-data width of face, and a host reads 8 bit data from the unique location to 4 billion, or it enables it to write it in. The interface of the R bus is carried out to a host computer through an address register, a data register, and a control register. A host interface bus controller constitutes these from the method of common use, and they are prepared all over the memory of a host computer, or input/output space. instantiation:1 of the function connected to R bus -- eight samplers which are one group who is made to period-ize a sampling clock and reads the value of the data sampled from other R bus locations according to the command of a host computer in case one location is written in through R bus.
2) Eight random access memory which is one group who changes a data value in case it writes in R bus location of specification [a host].
3) Layout section memory which is creating each memory location for the only R bus location. A host data access can be performed, and the layout section memory location on which the host was addressed can be read or written in by read-out or the write-in operation to an address space of R bus.
Such other functions can be thought out easily. Actuation of R bus is shown in drawing 42 . In order to read a location, the program which runs with the host computer which operates a rear riser system sets a "read-out" command bit to a host interface birth control register while loading the address to a host interface bus address register. Then, a host interface bus controller operates R bus read-out cycle. 8 bits of addresses are respectively given to each 1 time with the period of R bus clock at R bus data line. It expresses that R bus cycle has started the bus controller on the "synchronous" R bus controller line between the first cycle. Then, the period a "read-out" R birth control line and whose R bus clock are the 5th time can be accomplished, and addressed bus interface control logic block can complete the read-out operation. While R bus clock accomplishes the period which is the 6th time, addressed bus interface control logic block transmits read-out data to eight R bus data lines. A bus controller sets a "comp lied" command bit to a host interface birth control register while it catches this data and loads it to a host interface bus data register. The host program which recognizes a "comp lied" bit is set, and read-out and a "comp lied" bit are cleared for data. Except for setting a "write-in" command bit and loading the data which should be written in to a host interface data register, the writing of a location also has the same host program. A bus controller does not assert a "read-out" R birth control line in the 5th clock period, and transmits data to R bus data line in the 6th period. It is caught by bus interface control logic block by which data was addressed at this time. Bus interface control logic block constituted in a logic chip is equipped with the data path which connects R bus with a finite-state machine using the function controlled by the method which is used commonly completely according to the actuation mentioned above.
[0056] 1.5.3 The configuration section explains a configuration bus arrangement bus, and its use and operation. A bus is controlled by the host computer through a host interface. The interface of the bus is carried out to a host computer through a data register and a control register. Host interface hardware constitutes from the method of common use of these registers, and it prepares in the memory of a host computer, or input/output space. the configuration program which runs in a host computer -- **** -- the data loaded to a configuration bus data register is transmitted to a configuration bus data path. When a host computer writes in a configuration birth control register, host interface hardware makes a configuration bus clock expect 1 round.
[0057] 1.5.4 A controller and the reset controller of the reset rear riser system of a clock generation machine generate two reset signals. A system-reset signal is connected to all logic and the reset input pin of an interconnect chip. When a host claims, all chips are made into reset mode and it is made the preparatory state of a configuration. One or more programmable clock signal generators by idiomatic layout have the output signal distributed to the I/O pin of all L chips. A host can carry out [controlling the output frequency and stopping a cycle, carrying out a cycle again, carrying out the count cycle of specification,] a cycle continuously. A host is used as a clock generation machine of the layout section realized in a rear riser system. The operation of the layout section is controlled by controlling a clock signal. The reset signal of the layout section is connected to the I/O pin of all L chips. It uses as a means to reset the layout

section realized in a rear riser system in the reset signal of the layout section. These signals are applicable to connection with the layout section realized by the rear riser system. By including a special property in the network in an input design file, the circuit in the input design section is selected as a system reset or a clock. A layout section reader recognizes this property and a network is characterized as reset or the clock network of design-data structure. This network is assigned to the I/O pin connected to the layout reset signal or clock signal of hardware by interconnect of layout conversion system, and the network listing portion.
[0058] 2 Rear riser layout conversion system rear riser layout conversion system is equipped with a layout section reader, a base element converter, party SHONA, network listing and the interconnected system, the ERCGA netlist conversion tool, and the configuration file collector (drawing 43). Here, a configuration file and a correspondence table file are created as an output, using an input design file as an input. These are used for the various application for constituting and using rear riser hardware. In order to change an input design, :1 layout section reader is used, and the layout section is read into a memory data structure.
2) Change the base element in a layout section data structure into the logic chip base element which can be transmitted into an ERCGA netlist conversion tool and the suiting netlist file from the base element of a host EDA system proper.
3) Determine to which logic chip each component is used using party SHONA.
4) The netlist file to each logic chip and interconnect chip in a rear riser hardware system is outputted using network listing and an interconnected system.
5) Change each netlist file into a corresponding configuration file by repeating and using an ERCGA netlist conversion tool.
6) Using the configuration file collector which is an easy method, catch the configuration data transmitted to the single configuration file of this layout section, and rear riser hardware consists of configuration files of each logic and an interconnect chip using this.
The method for the layout conversion explained here is used for conversion of the combination of the logic gate of the input design section, and a flip-flop except for having been careful. The base element of the specific purpose is changed using the modification of these methods. These modifications are explained in the corresponding section.
[0059] 2.1 A layout section reader layout section reader constitutes the layout section data structure which corresponds while reading an input design section file.
[0060] 2.1.1 The input design file created by the requirement host EDA system of an input design file has the description about base elements and these I/O pins, and the description about the network which interconnects with the input/output terminal of the layout section while interconnecting two or more pins mutually. The input design file is equipped also with the information relevant to the base element and pin like a name, and a network. The input design file must serve as a gestalt of a base element so that rear riser layout conversion system can read. A "base element" is the gate, a flip-flop, or a fundamental logic element like a memory device. It is necessary to change into a configuration base element the configuration of the twist high level specified with the base element which a designer can specify by the EDA system in front of read-out of a rear riser system. As an example of the base element of the lot permitted in the input design section, there are the following Mentor Graplrics Quick Sim base elements. This is the easy gate (BUF, INV, AND, OR, NAND, NOR, XOR, XNOR) which has an input terminal to : read in a suitable example, and 25 pieces.
- It is the gate (DEL : delay element; RES : resistor; NULL : open circuit) specially.
- The nondirectional transmission gate which is a tri-state output (XFER)
- Storage device (LATCH, a level induction flip-flop or REG, flip-flop by which the clock was carried out)
- Memory device (RAM or ROM)
[0061] 2.1.2 A layout section data structure layout section reader constitutes a layout section data structure, changes a base element into the gestalt suitable for logic chip network listing using this, divides a base element into the partition of a logic chip size, and determines how a logic chip is interconnected. Moreover, finally a layout section data structure is read to the netlist file of each rear riser logic chip. The data structure consists of each base element of the

layout section, each pin, and the record of each network. Each record is equipped with the data about other entities to a record and links (namely, pointer) according to relation.
- A "base element" is the gate, a flip-flop, or a basic logic element like a memory device.
- Each base element is expressed by the base element record..A base element record has the link with other base elements while having the type and data about a base element like an object identifier.
- A base element record is a linked list at a duplex.
- A pin is input connection of a base element or output connection.
- The pin of a base element is expressed by a series of pin records which have data concerning [whether the pin name and the pin are reversed and] pins, such as an output drive of a pin, while it adjoins a base element record and is arranged.
- Each base element has only one output pin, and can make this the pin record of arbitration.
- A "network" is the set of the pin which interconnected.
- Each network is expressed by the network record. This network record is equipped with the link to other networks while it has data about a network like that object identifier.
- Prepare a network record into a linked list at a duplex.
- Prepare the pin of a network into the wrap-around list of a single link.
- Each pin record also has the link to the network of a pin.
- Each network record has the link to one pin in a network.

Drawing 44 a shows the easy example of a network, and drawing 44 b shows how a network is expressed using a layout section data structure.

[0062] 2.1.3 The methodology layout section reader of a layout section reader aims at constituting a correspondence layout section data structure while it reads the layout section which should be realized from an input design file. Explanation here is Mentor Graphics. It conforms to a layout file. The same is said of others. The layout file has the entry called an instance (instance) to each base element in the layout section. The information about the specific aspect of the base element attached in the instance in a layout file is the feature. The name in the parenthesis of each production process shown below is a name of an actual routine used in a suitable example.

1) As for each instance of the base element of :each layout file which creates the record of a base element, and the record of the pin in the data structure in memory over each base element in a layout file as follows, the type of :base element reads what it is (get dfi model type). When the information about arrangement of this base element that the user specified exists, this is obtained from "I chip" property. ; Search a more advanced un-fundamental instance using a layout file interface. This instance is equipped with this base element, and investigates a property similarly (get dfi 1chip). Each pin of an instance catches the property of the arbitration about pins, such as a name of :pin, (get dfipin info). To the following pin the record in a memory inside installation meter data structure -- this base element and pin -- assigning (alloc prim and pins) -- a base element record is filled. Each pin fills :pin record. (The object identifier number of the network where it connected in the layout file is memorized, and the number of the identifier of a truck is maintained to max.) To the following pin To the following layout file instance The table (net table) of a pointer is assigned to a pin record (pin pointer). An index is attached to the pin record to the each network which can be constituted by the object identifier number. It is referred to as NULL at first. A table is made according to the above-mentioned maximum identifier number.

2) Link the pin record of each network, and if :each pin record is made into the object identifier number of the network where : 'id' was connected in this pin in each following base element records in the data structure in :memory cyclically made into a linked list, when net table [id] has the non-NULL pin pointer, copy this to the "next pin" link of this pin record. The pin pointer to this pin is put into net table [id]. To the following pin To the following base element

3) Each pin pointer in :net table which creates the network record to each network as follows assigns :network record. A network record is connected to the pin which a pin pointer directs using a link. By addressing using an object identifier number, the information about a network is acquired from a layout file interface (dfi get net, get dfi net info). It directs on :network record

about each pin in the wrap-around list of the pin record in this network. To the following pin : which closes a wrap-around list -- the last pin is made to link to the first pin To the following pin pointer A net table memory storage function is canceled.
4) Internal-memory design-data structure is completed and display all the data about the layout section which the latter part of a layout section translation process needs and which should be realized.
[0063] 2.2 Base element converter base element conversion is Mentor Graphics Quick Sim. The base element in the layout section data structure from a base element like a base element which a host specifies is fitted with an ERCGA netlist conversion tool, and it aims at changing into the base element of logic chip assignment sent out into a netlist file. Some of this conversion is simple and direct, and it is only replacing only the type and pin name of a base element. Other conversion is quite complicated. The specific example of a citation shown below is a thing for a suitable example, and is Mentor Graphics. Mentor Graplics Quick Sim which exists in an input design file The base element which a host specifies, and the base element which a XilinxLCA logic chip specifies are used. When it has many input terminals rather than the gate of the layout section is permitted in the gate base element which a logic chip specifies, this gate is replaced in the network of the gate which has an equivalent function. Each of this gate circuit network has the input terminal of the number of permissions. In order to perform such substitute, the base element record and pin record of the gate are removed, and the new base element record of the gate and a new pin record, and the network record of the new circuit in a network are linked to the pin connected to the gate added and replaced, the pin record of a network, and a network record (drawing 45 a). When it has the function which the flip-flop of the layout section cannot use in the flip-flop base element which a logic chip specifies, a flip-flop is replaced in the network of the gate which has an equivalent function. A network is analyzed [1st] first and it investigates whether a function is always connected to the network which is not a fixed value. For example, when using the base element REG which a host specifies using both the direct clear input always connected to the activity network which is not a fixed value, and a direct set input, the base element in the layout section data structure in memory is replaced in the network of the gate similar to being used for the 747TTL flip-flop logic PERT who functions if needed. However, in the case of the AND gate which has one input terminal by which the direct set input was connected for example, to the grand network when [like a grand network] always connecting with the network of logic value zero, a direct clear chisel is actually needed, instead it is substituted for a logic chip D flip-flop. S A RAM base element is random access memory which has an address input terminal, a bidirection dataport, read-out enabling, and write-in enabling. A RAM base element is created in one or more rear riser layout section memory modules. Base element conversion software changes S RAM into one or more X RAM base elements which match an available layout section memory configuration directly. S A ROM (taking out exclusive memory) base element is the same as that of S RAM except for having added the file which an enabling input terminal does not exist and includes the contents of the ROM. S Change a ROM base element into one or more X ROM base elements which match a layout section memory configuration directly. Although X ROM has the read-out enabling input terminal, it does not have write-in enabling. The location to the path name of a contents file and S ROM of a basis is memorized using each X ROM base element. When it constitutes rear riser hardware using this layout section, while a configuration system takes out the contents of X ROM using a path name, these are loaded to layout section memory through a host interface. It has a separation I/O dataport. S RAM is operated similarly. However, this is Mentor Graphics Quick Sim. It does not prepare into a base element. The pin and network of the Original Engineering Consult. section show that the initial property, i.e., "inits", was transmitted and some initial value is transmitted continuously in a certain case. The continuous initial property which is a known value (0 or 1) is observed by the rear riser system, and it is suitable "gland" (namely, logic value 0) about a pin or a network. Or it connects with the network of "VCC" (namely, logic value 1). Specific Mentor Graphics In a case, the initial property of :, and T, X, R and Z is disregarded. Only OSF (=0=0S) or 1SF (=1=1S) is observed.
- Make an output pin into a part of gland or VCC network by 0SF or 1SF of an output pin of a

network, i.e., a network. [of arbitration]
- By 0SF or 1SF of an input pin, insulate this pin and connect with a gland or a VCC circuit.
[0064] The output pin of the Original Engineering Consult. section transmits the drive of various strength, and shows the type of the output structure which should be formed by the simulator. A rear riser system observes these strength a little in the case of base element conversion. The strength of the drive of an output terminal to the time of a high is zero, when it is a low and characterizes it as the strength of a drive being strength, an output terminal is identified as an open collector, and this is connected [just] to other same output terminals and registers with the gestalt which a logic designer calls "wye yard- and "network (drawing 45 b). Similarly, the strength of the drive to the time of a low is zero, the output terminal whose strength of a drive is strength when it is a high is an opening emitter, and it is used for forming "wye yard or ". If it finally does not enable, the output pin of a XFER base element does not have a drive, but is wired with other XFER output terminals and a register, and forms a "tri-state" network (<A HREF="/Tokujitu/tjitemdrw.ipdl?N0000=237&N0500=1E_N/;> =?<98 <8///&N0001=347&N0552=9&N0553=000049" TARGET="tjitemdrw"> drawing 45 c). All such structures are recognized by base element conversion system, and as the section of a tri-state network explained, it is changed into the logical-circuit network of the sum of the product which has an equivalent function. Specific Mentor Graphics In the case: The strength of a -X-State drive is disregarded.
- Although one or more XFER output terminals are connectable with a network, other output terminals are not connectable. As an exception, RES (resistor) which has connected the input pin to a gland or a VCC network is also connectable. When not enabling XFER, a network value is logic value zero, and in not connecting RES connected to VCC, it becomes the logic value 1. When enabling many XFER(s) from 1, a result serves as logical OR.
- 0C / 0E output terminal (SZ/ZS) can drive only the network which can be driven even if it uses the same driver. When not driving, 0 C networks become yes, it is disregarded whether RES is connected or not and 0E network serves as a low.
- Remove without an error the base element which has RZ, ZR, RSS, SR, or ZZ output drive.
- It is SZ or ZS, and SZ&ZS which have [: /, 1 / i.e., /, / which a fatal error produces according to the conditions of the following output circuit networks] many resistors, XFER and Strong, XFER and SZ, XFER and ZS, SZ that does not have the resistor or ZS, and Strong from many Strong (strong), Strong and resistors, and 1.
[0065] Mentor Graphics A special procedure for changing the base element of the suitable example which has a host and XilinxLCA is initial conversion (convert s to x) to the LCA base element of the base element which :1 host who is as being shown below (the name of a subroutine continues after each header) specifies. The base element which a host specifies is above-mentioned Mentor GraphicsQuick Sim. It consists of a set and a name is attached using the 'S' prefix. The base element which LCA specifies consists of Xilinx.xnf specification, and a name is attached using the 'X' prefix. In :S INV, each base element is replaced with X INV, and replaces the name of a pin. S In BUS, replace with X BUS and replace the name of a pin. S In the case of RES, replace with X BUF and RR drive and replace the name of a pin. S In the case of DEL, combine an IN&OUT network. S In AND, S NAND, S OR, S NOR, S XOR, and S XNOR, replace with X AND, X NAND, X OR, X NOR, XXOR, and X XOR, and replace a pin name. (An error is made when [than 25 pins] more) In S REG, it replaces with X DEF and a pin name is replaced. S In LATCH, replace by X DLAT and replace a pin name. S In XFER, leave the pin name even behind. S In NULL, delete a pin name. S In RAM or S ROM, leave the pin name even behind. To the following base element
2) Processing of an initial property (get init). Two networks in the layout section data structure in memory are special. : That is, they are "gnd" (logic value 0) and "VCC" (logic value 1). Each network is this network to a gnd network, and combination and the next network, when a gnd network cannot recognize that an initial property is OSF when the initial property of :network is OSF and it can recognize to the next network. When a VCC network cannot recognize that an initial property is 1SF when the initial property of a network is 1SF and it can recognize to the next network, it is this network to a VCC network, and combination and the next network. Each

output pin is this network to a gnd network, and combination and the next network, when a gnd network cannot recognize that an initial property is OSF when the initial property of :pin is OSF and it can recognize to the next network. When a VCC network cannot recognize that an initial property is 1SF when the initial property of a pin is 1SF and it can recognize to the next network, it is this network to a VCC network, and combination and the next network. To the following pin Degree network puts in :pin record during a list. The initial property of :pin is 0SF, and when this network is not a gnd network, each input pin separates a pin from a network, and connects it to a gnd network. The initial property of a pin is 1SF, and when this network is not a VCC network, a pin is separated from a network and it connects with a VCC network. To the following pin To the next network

3) Check all output pins, and while removing a base element without affecting the strength [a rear riser system] of a drive, remove XFER [DIZEBURU / always / XFER / which enable]. Each base element is to the following pin, when :output pin does not have Drives SS, RR, and SZ or ZS. An output pin is separated and removed when the output pin has RZ, ZR, RS, SR, or ZZ. When an output pin is S XFER: When E0 (enable) pin is always a low, delete a base element. When E0 pin is always a high, it substitutes for BUF. To the following base element

4) Sort out illegal multi-output connection, and identify and change wired OR, wired AND, tri-state networks, and these drivers (wired- nets). Each network puts in :pin record during a list. A XFER output pin, an input pin, and a non-XFER output pin are counted. these pins -- Strong (strong) -- and -- REJISUTIBU (resistive) -- SZ (open collector) Or it is ZS (opening emitter). In the case of the only output pin which does not have drive reinforcement or it has Strong, it is to the next network. When connecting one or more registers, it checks having connected all registers to either 'VCC' (pull-up) or 'ground' (pulldown), and memorizes any they are. Bigger Strong than :1 which exits by being an error in the following cases (exit), a bigger register than 1. XFER and Strong, XFER and SZ, XFER, and ZS. SZ, or ZS, SZ and ZS which have SZ which does not have the register or ZS, and Strong. In the case of 1 Strong and one register, the base element which has a REJISUTIBU drive is deleted. In the case of bigger SZ than 1: (open-collector wired AND) In the case of :register, each output pin checks that an output pin is pull-up, and deletes this. In not being a register, while making the drive of this pin ** into Strong, constituting X-INV and connecting this input terminal to an output pin, this output terminal is connected to a network. To the following pin A network is characterized as a "floating high" tri-state network, and interconnect constitutes this using OR/NOR gate. 1 -- case [of many ZS (s)]: (opening emitter wired OR) -- each output pin -- case [of :register]: -- a pin -- pulldown ****** -- things are checked and this is deleted after that. The drive of a pin is made into Strong when it is not a register. To the following pin A network is characterized as a "floating low" tri-state network, and an interconnector constitutes this network using the OR gate. 0 -- many XFER(s) and register nothing, or case [of PURUDAUN]: (tri-state "a floating low") -- every -- S-XFER -- :AND XFER which constitutes I0 E0 (or ENA) and AND S-XFER is changed into X-AND using XFER10 which constitutes I1. To following S-XFER The register base element of arbitration is deleted. A network is characterized as a "floating low" tri-state network, and interconnect constitutes this using the OR gate. XFER which constitutes [many XFER base elements and in the case of pull-up] :NAND10 from 0 in the case of a :(tri-state "floating high") 1 piece S-XFER base element Using XFER10 which constitutes E0 (or ENA) and NAND11, S-XFER is changed into X-NAND and it is reversed. In the case of the S-XFER base element which is many from 1: Each S-XFER is XFER which constitutes :AND10. Using XFER10 which constitutes E0 (or ENA) and AND11, S-XFER is changed into X-AND and it is reversed. A register base element is deleted to following S-XER. A network is characterized as a "floating high" tri-state network, and interconnect constitutes this using OR/NOR gate. To the next network

5) The gate of the arbitration which has many input pins is replaced rather than it approves in the gate base element which LCA specifies using the gate circuit network which has an equivalent function. Each of a gate circuit network has the input terminal of the number of permissions. Each base element: (wide-gates) It is the case (it is assumed that it is a thing using XC3000 logic chip) where there are more the gates and input terminals than 5, and they are 25

or less. : the final output gate of the same class is constituted. The output terminal is connected to an original output terminal and a copy property. Each of the smaller input gate needed : (using AND for AND or NAND original) The gate is assigned. The output terminal of the gate is connected to the last gate input terminal. The input terminal of the gate is connected to an original input terminal. To the next gate The original wide gate is deleted. To the following base element

6) while checking the function of a flip-flop -- constraint of LCA -- ****** -- arrange like. When using XC3000 series, although a flip-flop has a direct clearance, it does not have a direct set, therefore does not have both. since all S-DEF has the set and the pin for being clear, if there is no flume which has the pin, it is alike, and it does not need to be concerned but it is necessary to replace a base element Since XC3000 does not support a latch, it needs to replace a latch with an equivalent gate circuit network (flops_for_3K). When :base element of each base element is DLAT or DEF: While memorizing each pin, separate. When SD and RD network confirm whether to be 'ground' or 'VCC' directly or in un-direct through the gate, it finds out whether SD and RD are always lows. When each base element is DLAT: Constitute the latch who has a nest all over the network of the gate, and has the gate for SD and/or RD only when required. An original base element and a pin record are deleted. When each base element is not DLAT but DEF: In a low, SD constitutes X-DEF, without using SD, and always connects this. When not a low but RD is a low, it uses X-INV for an input terminal and an output terminal, constitutes X-DEF, and it connects this, and SD connects RD pin of X-DEF to SD network. When SD always is not a low, they are a nest and TTL7474 to the network of six 3-inputs NAND and 2INV(s). DEF which has a set and a clearance is constituted similarly. An original base element is deleted. To the following base element

7) Change S-RAM and S-ROM into X-RAM and X-ROM. When :base element of each base element is S-RAM or S-ROM: Determine the height (height) (numbers of words) by counting the width of face of an address pin (height = power of 2 - a pin count), and ADOREPIN equal to a data pin number. The layout section memory configuration in which each use is possible carries out division of the :S-RAM/ROM height with layout section memory height, and obtains the line count of a required module. The number of trains of a required module is obtained by carrying out division of the width of face of S-RAM/ROM by the width of face of layout section memory. The total of a module required for this configuration is line credit ****. To the next configuration The configuration in which the required number of modules serves as min is chosen. When you need many modules rather than a line, it constitutes the base element and network of a decoder using the input terminal connected with the output terminal to each modular line in the address network of high order. Each line: (only X-RAM) The :2 piece input terminals which constitute the AND gate for write-in enabling using two input terminals are the decoder output terminal of this line, and S-RAM write-in enabling. : which constitutes the AND gate for line read-out enabling using two input terminals -- this two input terminal is the decoder output terminal of this line, and S-RAM lead enabling. To the following line About each modular line, :each column memorizes the configuration while constituting :X-RAM/ROM base element. In X-ROM, the file name, a line count, and the number of trains are memorized. About read-out and the write-in enabling pin of X-RAM/ROM, it is this line (only X-RAM). It is connection (or this line one case the enabling pin of S-RAM connection) to read-out and a write-in enabling pin. It carries out. The address pin of X-RAM/ROM is connected more to the address network of low order. The data pin of X-RAM/ROM is connected to the data pin of the lot corresponding to this train. To the following train To the following line An original S-RAM/ROM base element is deleted. To the following base element

[0066] 2.3 Party SHONARIA riser hardware consists of the hierarchy of a unit and a subunit. That is, a board is argument. It is that have a ** chip, a box is equipped with a board, and a rack is equipped with a box etc. Each unit has the capacity for interconnect with the logic of a unit proper, and other units. the layout section which should be realized is segmented according to this hierarchy (namely, sub division -- carrying out), and it considers as the multiplex cluster of a base element. The box partition of the lot created according to the logic and connection capacity of each box is prepared. It divides into such a small partition that it is enough to

program each of these partitions to a single logical chip by dividing into the sub partition of a board etc. Sequential application of the same division method is carried out at each level of a hierarchy. The purpose of division is making the total of holding the number of circuits linked to a partition, holding the amount of the logic used for division in the limit of three units, and 4 partitions, i.e., the total of a unit used, into min under assigning :1 each base element to a box, a board, and a logic chip and the interconnect capacity of two units (a box, a board, or logic chip).
[0067] 2.3.1 the division method -- the number of "cut (CUT) networks" (connection of the base element in the exterior of a cluster) is min, and the suitable division method explained here is based on the process in which the logic base element which interconnected by high density in each other is clustered. Each cluster is a partition corresponding to a box, a board, or L chip. Said process is Palesko accompanied by considerable amelioration pointed out to below. And Akers It is based on the conventional segmentation method (Chet A.Palesko, Lex A.Akers, "Logic Partitioning for Minimizing Gate Arrays", IEEE Trans, CAD, NO.2, pp.117 -121, and April 1983). The "Naru (null) cluster" which consists of the base element currently assigned to the cluster which is visible from the start in no base elements is prepared. First, a seed (seed) cluster is chosen from a NARUKU raster, this is repeated after that, the "advantage" of all NARUKU raster base elements is calculated, and each cluster is formed by choosing the base element which has the biggest advantage. It becomes a thing suitable for including in a logic cluster as the advantage of a base element becomes large.
[0068] 2.3.2 An advantageous functional division-advantage is based on how the number of the cut networks of this cluster changing, when building this base element into a cluster. In order to hold a unit below in the maximum interconnect possibility, it is necessary to count the cut network total of a cluster. Each network equipped with the pin of a base element has become perpendicular, and if the thing incorporating a base element is assumed, it will be classified into either a closed circuit network, the network of an "a large number cut" or the network of a "single cut." When only one connection is prepared inside a cluster, it is a single cut network, and if many connection is prepared inside a cluster from 1, it will become an a large number cut network. A closed circuit network says the network where the whole is contained in a cluster. If the base element which attached the shadow is moved into a cluster, drawing 46 shows what becomes, while showing a cluster and five base elements connected by three networks S, M, and E. If the one number of cut networks of a cluster is made to increase, Network S will turn into a single cut network, and if the one number of cut networks is decreased, Network E will turn into a closed circuit network. Even if Network M increases and decreases the number of cut networks of a cluster, they are cut networks, and for this reason, it is disregarded. [many]
Change of a cluster network is the difference of a single cut network and a closed circuit network. Namely, :[change of cluster cut] = [single cut network]-[closed circuit network]
A desirable advantageous function is determining whether it being optimal choosing which base element, in order to define the number of each base element and to include in a cluster. It is optimal to choose the base element most firmly connected with the pin of the maximum number. This function is Palesko andAkers. It is the first advantageous function about division. namely, : -- case [of [change of cluster cut] >0]: -- number of pins][of a [advantage] =[base element]/ [change of a cluster cut]
[change of a cluster cut] -- case [of <=0]: -- [advantage] =[-(change of cluster cut) *100] +100+ [the number of pins of a base element]
When building this base element into a cluster, the number of cluster cuts increases. The more there are few cut networks added the more the more it has many pins, the more it excels. When the number of cuts of a cluster decreases, it has guaranteed that a bigger advantage than the advantage of what kind of base element which the degree of reduction becomes [advantage] 100 times, and 100 is added [advantage], and does not decrease a cut is acquired. When the cluster cut is decreasing, they are [cut / carry out and / the increase of a number, or / of a pin / connection] further many focus ********** about a base element. The amelioration used by the suitable method is adding the term of the number of pins to the ratio of the number of pins / cut change, when a cluster cut increases. This modification can improve initial seed selection by choosing the base element which has more pins, when said ratio is equal. By

doubling said ratio ten, the number of pins independent twist is also made effective. This is a suitable and advantageous function. namely, : -- case [of [change of cluster cut] >0]: -- [advantage]=([10*(number of pins of base element)]/[change of cluster cut])+ [the number of pins of a base element]
[change of a cluster cut] -- case [of <=0]: -- [advantage] =[-(change of cluster cut) *1000] +100+ [the number of pins of a base element]
[0069] 2.3.3 Arrange all base elements in a NARUKU raster at the beginning of the configuration of a cluster. A user can arrange a base element beforehand in a specific cluster by adding the property which displays L chip to choose, a board, etc. to the input design section. At this time, these base elements arranged beforehand play a role of seed arrangement which carries out a function to cluster information. By this, a user can change the result of division by collecting other base elements which can collect a timing sensing base element or other high priority base elements, and are firmly connected to the high priority base element. About each new cluster, first, the advantage of the base element which is not arranged [of a new cluster] is calculated, and it records into a base element record. When not preparing arrangement beforehand, the most advantageous base element (namely, thing which has the biggest advantage) is chosen as an initial seed base element of a cluster. After moving each of the most advantageous base element into a cluster, only the incorporated base element and the base element which has a pin in the same network calculate an advantage again. Since other base elements are not influenced by migration, there is no change in these advantages of a cluster. Therefore, the most advantageous new base element is moved into a cluster until a cluster fills. It is dependent on both logic capacity and interconnect (namely, cluster cut network) to determine the time of a cluster becoming full. When moving a base element into a cluster, the gate number in a cluster always increases with a base element. However, a cut network does not necessarily increase with a base element. It can decrease. Palesko and Akers Although the base element with which a base element does not exceed logic capacity or the limit of interconnect and which has advantages fewer than max can be moved into a cluster if it becomes when arriving at the limit of interconnect by the method, when exceeding the max of local interconnect, a base element cannot be moved into a cluster. The method explained here is improved in the following points. namely, : -- a marker's (marker) array is prepared. A marker can start to each marker. It moves one base element at a time into a cluster. The number of cluster cut networks is checked after each migration. When the number of cluster cut networks is below the maximum available interconnection function of a unit, migration is recognized as what can interconnect. When the limit of the maximum logic capacity is arrived at, the last migration is not recognized to be what can interconnect, and migration is not performed until interconnect of the last migration is attained. In order to divide a unit (a rack, a box, or board) into a subunit (a box, a board, or L chip): Move all the base elements that are not arranged beforehand into a NARUKU raster. Each cluster is memorized while it calculates the advantage of :each NARUKU raster base element. The number of starting counters is made into zero. [Cluster base element count] In < [the maximum logic capacity], a migration counter is incremented. The most advantageous base element is moved into a cluster. It records on a migration counter which base element is moved. [Cluster cut network] In < [the maximum interconnect capacity], move [migration counter] =O.K. is marked. [Cluster cut network] In >= [the maximum interconnect daily dose], it is move [migration counter] =NOT. O.K. is marked. The advantage of the network connected to this cluster is calculated. To the next repeat move [migration counter] =NOT In O.K., the base element recorded on move [a migration counter] is moved outside from a cluster.
DEKURIMENTO [a migration counter]. To the next repeat To the following cluster The process of segmentation continues until it arranges all base elements with the sufficient result in a cluster, or until all clusters become full, and a process ends it. In order to segment all the layout sections of a suitable example, it divides into one cluster for every box of :latch level, i.e., a box. Under the present circumstances, :[maximum logic capacity] =[all box and maximum interconnect capacity] = [the Y-Z pass for every box]
**********. Each box cluster is divided into one cluster for every segmentation to the board of :box level, i.e., a board. Under the present circumstances, :[maximum logic capacity] =[all

board and maximum interconnect capacity] = [the X-Y pass for every board]
**********. To the following box cluster Each board cluster is divided into one cluster every segmentation to L chip of :board level, i.e., L chips. Under the present circumstances, :[maximum logic capacity] =[L chip and maximum interconnect capacity] = [the L-X pass for every L chip]
**********. To the following board cluster
[0070] 2.3.4 Depend for the decision of the limit of the maximum logic capacity of being used for the marginal this gentleman method of capacity on the property of the logic chip to be used. When using LCA made from Xilinx as a logical chip, these are the logical blocks (CLB) which can be constituted. It is made the base. Many the gates and flip-flops can be embodied by each of CLB. It depends for the number of CLB(s) on of which grade many the gate and a flip-flop function, and a function are prepared, what pin many it has, or how it interconnects. When changing the layout section into the gestalt of CLB before division, it considers as the base element which had CLB divided, and the limit of logic capacity is based on the number of CLB(s) in LCA. When not changing the layout section into the gestalt of CLB before division, the limit of logic capacity is based on the number of the gates which should suit LCA as a base element divided in the gate. According to the degree of the capacity which the gate consumes, the weight of the gate is applied and the result of division is improved. The limit of being used for constituting each cluster does not necessarily need to be altogether the same. When changing logic and interconnect capacitance characteristics between units, a limit is set up appropriately and the cluster of these units is constituted.
[0071] 2.3.5 By dividing the segmentation process of a rear riser, 3 number box / board / chip location to each base element of the layout section are obtained. This location is memorized by the base element record of a layout section data structure. By this, each base element of a layout section network can be continued and traced in L chip, a board, and a box. The timing of a network is evaluated by adding delay through an interconnect crossbar chip and a logical chip while tracing the network in a system. In the phase of interconnect, order of the netlist is carried out based on the combination total of various box / board / chip base elements which it had all over the network. Thus, interconnect guarantees from the most complicated network to the least complicated network. Finally, since it has the information to which the base element of a network and a network record covers L chip and a crossbar chip, and creates a network clearly, it is only necessary to segment a local diagrammatic logic change again and to update a chip equipped with the changed network. The layout section can be transformed by this, without segmenting the layout section again.
[0072] 2.4 Network listing, interconnected-system RIARAIZA network listing, and interconnect conversion system aim at constituting the netlist file about each logic chip and crossbar chip in a rear riser system which are used for constituting rear riser hardware according to an input design. That decision which should carry out the netlist of the partial crossbar interconnect how is made by synthesis of the following three-step process.
Stage 1: A statement is transmitted to the logic chip netlist file about all the logic base elements in design-data structure for every base element.
Stage 2: The statement about the addition gate of the tri-state network completely included in the single logic chip is transmitted for every network.
The netlist of the interconnect of the network which passes along between many logic chips from stage 3: is carried out. The statement about all the interconnect buffers of this network in all chips and the addition gate of this network in a crossbar chip is sent out for every cut network. As a part of this process, it is determined clearly how a network is interconnected. This process itself has four stages.
Stage 3a: Constitute the tree (tree) in which it is shown a passage and how and where a network stations a logic chip driver and a receiver for between each crossbar.
Stage 3b: Evaluate the capacity which interconnects the network of each class of a crossbar chip.
Stage 3c: Choose the optimal group of the crossbar chip which interconnects this network.
Stage 3d: Carry out the netlist of the interconnect based on selection and the tree structure of a group by sending out the statement about a buffer and the addition gate to logic and a

crossbar chip netlist file. With this section, the technology used for each stage is explained and two detailed examples of circuitry are described to be detailed conventions about perfect interconnect and network listing procedure.
[0073] 2.4.1 the interconnect structure of a simple network and a tri-state network -- a simple network is a network which has only a single driver. The source L chip which has a driver transmits a signal to the crossbar chip which has attained to all receivers toward a hierarchy's upper part. A hierarchy goes caudad, the pass for driving a receiver is connected, and all receiving L chips are driven. Drawing 47 shows interconnect of a simple network and, for details, explains below. A tri-state network is a network driven by two or more tri-state, the open collector, or the opening emitter driver. This is shown as a single network which has two or more drivers (output pin) in a layout section data structure. They are the one AND gate and one or more receivers (input pin) which change a driver and are obtained between base element conversion of each driver. The "floating low" network which serves as zero when not enabling a driver is realized by driving the one or more addition OR gates by the AND gate. The "floating high" gate has the reversal data input in the AND gate, and is setting the final addition gate to NOR. The same topology and a fundamental method are applied to both cases. While it is general, a tri-state network is materialized as the sum of a product using tropism connection and the one or more addition OR gates. The pass of a driver is centralized toward Z from an interconnect hierarchy's X, and a driver is brought together in the addition OR gate. The highest addition OR-gate output of level is made into the true value of a logical-circuit network, i.e., the source. An interconnect hierarchy goes caudad, the source is connected and all drivers are driven. As a result, some chip pairs (Z-X, X-Y, and/or Y-Z) need two pass. Among those, one pass has connected the driver with the addition OR gate, and connects a receiver and an output. Drawing 48 shows interconnect of a tri-state network, and, for details, explains below.
[0074] 2.4.2 Specify the interconnect in a logical chip into a netlist file using the network which has the name of a naming proper. These networks should not be confused with the network in a layout section data structure. Each layout section network has network of one of the two in a logic logic chip netlist file, and uses the same actual network name as having used for the input design file for a netlist file. The name generated artificially is attached to the network added to a layout section data structure between base element conversion. The network which does not exist in a layout section data structure is sent out to a logic chip and a crossbar chip netlist file, and interconnect is specified. A logic chip or crossbar chip When using the network between an I/O buffer and an I/O pin, the network between the AND gate and the addition gate which is the sum of the product of tri-state, and crossbar addition, although all the upper part of interconnect and the network along which it passes caudad, and these networks are connected with the single network of the layout section, its network in a netlist file is separate. In case an interconnect base element is sent out to a netlist file, an actual network name is changed and a separate network name is offered to each of these interconnection functions. The following chart is listing all the name modification. Only one name is used for every chip level between an I/O buffer and its pin. According to the chip in the other end of connection, a number is given to these names, and peculiarity is given to them. The name used for every chip level prescribes crossbar chip internal connection. [than 1 time] [more] This is only an example of the naming system which such many can constitute. An alphabetic character 'H' is used instead of the actual network name in a chart. For example, when calling 'ENABLE' the network which interconnected, the network between the input-buffer input terminal received from the logic chip 6 and its I/O pin is called 'ENABLE-D -6'.
'N':L chip: When using this L chip as the source of a network, it is a true network value. When preparing the driver of one ** in this L chip, it is a tri-state driver.
X, Y, Z chip: When preparing one Childe (child) driver, it is an input-buffer output pin from Childe. When using this chip as the source of a network, it is an output-buffer input pin to Childe.
All chips: It is an output-buffer input pin to parents. Addition gate output terminal.
'N_R':L chip: When preparing the source of this network in either, it is a true network value.
X, Y, Z chip: When this chip is not the source of a network, it is the input pin of Childe's output buffer.

All chips: The input-buffer output pin from parents.
' -- N_R_c': -- they are X, Y, and an output-buffer output pin to Z chip:Childe. 'c' is Childe's chip number here.
'N_P': The input pin of the input buffer from the chip:parents who are all.
'N_D': The output pin of the output buffer to the chip:parents who are all.
' -- N_D_c': -- X, Y, and the output pin of the output buffer from Z chip:Childe. 'c' is Childe's chip number here.
'N-P': The input pin of the input buffer from all chip parents.
'N-D': The output pin of the output buffer to all chip parents.
' -- N-D-c': -- the input pin of the input buffer from X, Y, and Z chip tea IRUDO.
'N_OR_i':L chip: When many drivers are prepared in L chip from 1, it is a tri-state driver. 'i' is identifying many of such drivers here.
X, Y, Z chip: When preparing many Childe drivers from 1, it is an input-buffer output pin from the Childe driver.
All chips: Addition gate input terminal.
[0075] 2.4.3 Stage 1 : the network listing statement of a logic base element is sent out to the logic chip netlist file to all the logic base elements in a layout section data structure for every base element. The network which has connected the base element is named and it is made in agreement with naming used for a stage 3d [following] interconnect buffer. An input pin is connected to these original network names when preparing the source of a network into the same logical chip. In case this is always truth and drives L chip of a cut network to a closed circuit network (network without a cut), it is truth. An input pin is connected to these parent receivers' input buffer when not using L chip as the source. An output pin is connected to these original network names except for the case where an output pin is connected to the addition gate of a logic chip. When connecting an output pin to the addition gate of a logic chip, a specific network name is made to change.
[0076] 2.4.4 Stage 2 : send out the statement about the addition gate of the tri-state network included in the network listing completeness of the logic chip addition gate in the single logic chip for every network. An input terminal is connected using modification of the network name mentioned above. An output terminal drives an original network name. According to whether a network is a "floating high", suitable output detection (OR or NOR) is used.
[0077] 2.4.5 Stage 3 : carry out the netlist of the interconnect of the network (cut network) which passes along between many logical chips from the decision and the network listing 1 of cut network interconnect. A cut network is respectively processed at once through Stages 3a, 3b, and 3c.
[0078] 2.4.5.1 Stage 3a : constitute a tree data structure [-like at the time of style Seiichi of an interconnect tree], and guide an interconnect process. This temporary tree data structure expresses the structure of a network by indicating the requirements for each interconnect to be L chip which has a base element to this network, and X, Y and Z chip which materialize interconnect. :level which memorizes as follows the data about the interconnect pass which each node in each tree level supported the logic or the crossbar chip in a system, had the branch connected to the tea IRUDO node of the low order, and has led to the pass of a node and parents Chip Interconnect pass root Z chip The nothing 1st level Y chip The 2nd level of Y-Z pass X chip The 3rd level of X-Y pass L chip However L chips each included all over the L-X pass network may have many base elements to the network, they will be expressed by the node of the only individual in a tree. Each node has the following entries.
Chip number: One L chip of the boards, one board of the boxes, or one box of the racks. Initial value is NULL.
D and R count: It is the number of the drivers (D) and receivers (R) which are needed for the pass of this node. Initial value is zero.
D and R pass: (from the inside of some pass numbers which can be used with each L-X, X-Y, or Y-Z pass) Using one of pass numbers, a driver goes up a tree from this node, and a receiver gets down. Initial value is NULL.
Top sum: When this node has the addition gate which equips low order with all drivers, recognize

it as truth. Using this, the last gate in the sum of the product of multigate is controlled, and, in a "floating high", the output reversal is obtained. Initial value is a false. When the network has not reached many boxes, the root node (root node) has the 1st level node of NARUENTORI and one **. When many networks have not attained to a board, the 1st level node has the 2nd level node of NARUENTORI and one **. When the network has not attained to much L chips, a network does not have the necessity for interconnect and does not have the tree. According to the location of the base element assigned by party SHONA, the network in a layout section data structure is operated, and a tree is constituted. When the network has not attained to many boxes or boards from 1, the node of the crossbar level which is not needed is made into Naru. Thus, while counting the number of the drive output terminal of L chips each, and receiving input terminals, it memorizes in L tip node, and the need for interconnect of L chip is checked. In X tip nodes each, the number which has the number and receiver of L chip which have the driver is counted, and if X chips each do not prepare what kind of interconnect, they learn, and they check **. Similarly, in Y chips each, a drive and a receiving X chip are counted and Y chip is counted in Z chip. Finally, a tree is analyzed and the point which transmits the true value of the network which is the source to a receiver is determined. In the easy network, the source is prepared into one L chip. Since crossbar addition is used, the source can also be considered as the crossbar chip of a tri-state network. Usually, when the crossbar chip has the receiver between tea IRUDO chips, the netlist of the crossbar chip is carried out and true value is transmitted from the manager of a youth hostel chip of the twist high level. However, when the low-ranking chip has the source from the parent chip in the hierarchy, a parent chip acquires true value from a low-ranking chip from the parent chip itself or it. In order to do in this way, a crossbar node is scanned, and a receiver count is set to zero when a descendant of a node or a node is the source.

[0079] 2.4.5.2 Stage 3b : since decision Z chips each of the interconnect capacity of each set connected the same Y chip in each box and Y chips each have connected the same X chip of each board, form a lot with X, Y, and Z chip which interconnected. 64 sets is prepared in the suitable example of a rear riser system. The each is equipped with one Z chip, eight Y chips which have the Y-Z pass using Z chip prepared all over [one] each box, and 64 X chips which have the X-Y pass using Y chips each prepared in each one board. This is permitted although the pair of each group has the same X chip in this case. The reason is that only the group of one ** is chosen and it interconnects a network. a group which calls pass a pair of each of the chip which interconnected like L chip and X chip -- it connects with a wire. The pass in each crossbar is listed on a pass table. The L-X pass table has the element relevant to each pass of each L-X crossbar in the whole system. A L-X crossbar is prepared in each board in each box, and the pass related to L chips each and X chip of a lot is prepared in each crossbar. Thus, the L-X pass table has the size of five pieces. That is, they are LX [a box], a [board], [L chip], [X chip], and [pass]. Similarly, a X-Y pass table, i.e., XY [a box] and a [board], [Y chip], [pass], and a Y-Z pass table, i.e., YZ [a box] and [Z chip], and [pass] are prepared. Each element in a table is made into free ["free / free (free) /"] or "YUZUDO (used)" in interconnect procedure. A table element is used when the input or output I/O pin sent out to the netlist file uses pass. It determines by catching the free pass count to each pass which should interconnect the network interconnect capacity of each class. The Y-Z pass between Y chip in a box and Z chip is first considered to the 1st. In each box in a network, the number of the free passes in the Y-Z pass table about Z chip in this group and Y chip of this box is counted and memorized. The X-Y pass between X chip on a board and Y chip in a box is considered to the 2nd. That is, in each board in a network, the number of the free passes in the X-Y pass table about Y chip of this box of this group and X chip of this board is counted and memorized. The L-X pass between L chip on a board and X chip is considered to the 3rd. That is, in each logic chip of a network, the number of the free passes under L-X pass about this L chip of this group and X chip of this board is counted and memorized. In any points, when enough free passes to complete interconnect do not exist, this group is recognized to be failure and this process is advanced to the following group. The pass count about each class of a crossbar chip which can attain interconnect with each sufficient pass under interconnect, i.e., the result, as a result will be caught.

[0080] 2.4.5.3 Stage 3c : since it can interconnect using the group of many selection of a group, hold the balance of the pass which chooses and uses one of a group. Development of a perfect interconnection function is guaranteed by this. An easy group selecting technic is choosing the group all whose pass counts' are maxes. However, this has disregarded local conditions. It is desirable to choose the group which has the biggest minimum pass count out of the pass count in all level. For example, if 2 sets assumes that it is what has the following pass counts Pass: YZ YZ XY XY LX LX LX group A:4 4 4 3 1 3 4 set B:3 3 3 3 3 3 3 set A The maximum total (23 to 21) Although had, choosing this means adopting the L-X pass from one L chip-X chip pair which can be used at the end. Group B is the biggest min (3 to 1). It has and an L chip-X chip pair is not closed. The group which has the biggest min is chosen until in association it eliminates one min from each class and chooses one group as a result of examination. (in the case of the 1st network) One is adopted when the same [all groups are actually the same, and]. This is the method currently used. Especially consideration is required when adding examination about the tri-state network of a lot. It is the pass for the input terminal with which one side is connected with the addition gate toward a hierarchy's upper part. Since some chip pairs must have these [which are used for the same network] two pass, the group chosen in these cases must have at least two free passes. Such a case is detected when the tree node (namely, X tip node for L-X pass etc.) of pass has D of non zero and R count, and the parents of non NULL.
[0081] 2.4.5.4 Stage 3d : carry out the netlist of the interconnect by giving selection and the tree structure of the network listing group of interconnect, and sending out the statement of a buffer and the addition gate to logic and a crossbar chip netlist file. For every level, this is first performed about a logic chip and is performed about X, Y, and Z chip after that. Interconnect and the directivity of each chip are determined by using the data in a tree. The netlist of each connection is carried out by sending out the statement about the buffer and network of interconnect to a netlist file. (When a tea IRUDO chip exists) The netlist of the connection between a chip and a tea IRUDO chip is carried out first. Each tea IRUDO chip is considered in order. When it is shown that the tree is making this chip drive, the netlist of the input buffer is carried out using the pin number which has connected the driver of a tea IRUDO chip. When this chip has many drivers from one piece, another network name is used for each. Thus, these network names are caught by the addition gate by which a netlist is carried out to behind. When the tree shows that Childe has received this chip, the netlist of the output buffer is carried out using the pin number which has connected the receiver of a tea IRUDO chip. When this chip itself is a receiver from those parents, this chip connects a parent receiver using a different network name. When this chip is equipped with many drivers from one into that Childe, the netlist of the addition gate is carried out and the driver circuit network mentioned above is connected. Finally, the netlist of the connection with a parent chip will be carried out (if it exists). When the chip or the descendant chip of arbitration is equipped with the driver, while adopting the interconnect pass about a driver from the pass table entry about the pair of a chip, and the selected group, the netlist of the output buffer is carried out and parents are driven through the pass which adopted here. When this chip is a receiver from a manager of a youth hostel, pass is chosen from a pass table and the netlist of the input buffer is carried out using this pass.
[0082] 2.4.6 detailed convention: about interconnect and network listing procedure -- : which prepares four classes about the 1st general convention:network -- a simple closed circuit network:network has one driver, and prepares all base elements into the same L chip.
A simple cut network: A network has one driver and prepares a base element into much L chips.
Tri-state closed-circuit network: From one piece, a network has many drivers and prepares all base elements into the same L chip.
Tri-state cut network: From one piece, a network has many drivers and prepares a base element into much L chips. : whose 'source' of a network is a chip which transmits the actual logical value -- in a simple network, the source is an L chip which has the driver. In a tri-state network, the source is a chip which has the top MOSUTO (top-most) addition gate. In order to determine this, :network is scanned, and it investigates where the output pin is arranged. When all output pins exist on the same L chip, this L chip is the source. When all output pins exist on the same

board in a case of other than [this], X chip on this board is the source. When all output pins exist all over the same box in a case of other than [this], Y chip in this box is the source. In other than the above, Z chip is the source. The index number of an output pin begins from the pin which it shows which output pin in the wrap-around list of the pin in the network it is, and a network record shows, and it counts it one [at a time] from zero.
Stage 1: Send out all the base elements in a layout section data structure. L chips each in a layout section data structure open the netlist file of this L chip, when not opened by the netlist file of :L chip. Each base element of this L chip sends out :base element header statement to a file. Each pin of this base element: (using the object identifier of the network for obtaining a name from an input-design file) Obtain the connected name of a network. And 'N' It calls.
In the case of an input pin: It is a network 'N' when this L chip has the source of a network. The statement about the connected input pin is sent out. It is a network 'N_R' when that is not right. The input pin statement which can be set is sent out.
In the case of an output pin: Obtain the index number of this output pin and call 'p'. In the case of a simple network, directions are taken out to the pin of a network 'N'. In the case of a tri-state closed circuit network, directions are taken out to the pin of a network 'N_OR_p'.
In the case of a tri-state cut network: It is a network 'N' when this is the output of one ** about this network of this L chip. Directions are taken out to a pin. When it is not a **** ** output, directions are taken out to the pin of a network 'N-OR-p'. To the following pin To the following base element To the following L chip
stage 2: -- : which takes out directions to all the closed circuit network addition gates -- each tri-state closed circuit network obtains the name of this network called : 'N'. This is opened when not opened by the netlist file of this L chip. It counts how many output terminals exist in the network called 'i'. : which sends out the statement about 'i' input gate -- when this network is a 'floating high', it is NOR, and in other than this, it is OR and has the input terminal connected to the network 'N_OR_j' (all j of zero to i-1), and the output terminal connected to 'N'. To the next network
: which takes out directions to all the cut network addition gates while taking out directions to a Stage 3:cut network and the buffer which interconnects -- all the elements of all interconnect pass tables are made "free". Each cut network (simple or tri-state) chooses the first biggest network within the limits of this sequence while it chooses a cut network in order of :hierarchy and chooses the 1st box network etc.
stage 3A: -- each base element of the configuration network of a tree -- : -- 1 is added when the tree node is not prepared in the box of this base element. 1 is added when the tree node is not prepared in the board of the base element in this box. 1 is added when the tree node is not prepared in L chip of this base element on this board in this box. When network connection of this base element is an output pin (namely, dry BIINGU (driving)), D count of the node of this L chip is incremented. In a case of other than [the above], when this L chip is not the source of this network, R count of the node of this L chip is incremented. To the following base element If only X tip node of one ** is prepared in case a tree expresses all the base elements of this network, Y tip node will be set to NULL. (That is, the network exists on a board.)
When only **** ** Y tip node exists, Z tip node is set to NULL. (A network exists all over a box.)

each non-NULL crossbar level -- setting -- first -- X chip, an after that Y chip, and after that Z chip: -- each node in this level -- :D= [the number of the tea IRUDO nodes whose D counts are not zero]
R= [the number of the tea IRUDO nodes whose R counts are not zero]
This node is set to R= 0 when using this node or descendant as the source of this network. When using this node as the source and making a network into tri-state, a that "top sum (top sum)" flag is set truly. To the following node To the following level
Stage 3B: Decision each class of the interconnect capacity of each class catches the pass count of each pass which should interconnect, and determines the interconnect capacity.
: which assigns storage of the pass count of this group -- Y-Z pass count: -- allocation X-Y pass count [of each box]: -- allocation L-X pass count [of each board]: -- : when preparing

only the box of an individual all over this network uniquely [of L chips each / allocation] -- the Naru (it is not zero) Y-Z pass count of this box is left as it is. When that is not right, :each box counts the number of the free passes in :pass array.
YZ [this box], [this group], [pass]
The tree node of this box has the parents of non NULL, and when it is D> 0 and R> 0, the pass of this box is "double." That is, it has both the driver and the receiver. When there are few free passes than 2, this group cannot connect this network. When it is except the above, and when pass does not exist, this group cannot connect this network. When this group is not connectable, this is made unusable and it advances to the following group. When it can connect, a total is saved as a Y-Z pass count of this box. When only the only board is prepared in this network,: (Y-Z interconnect is not needed) Leave the Naru (it is not zero) X-Y pass count of this board as it is. When two or more boards are prepared: Each board counts the number of the free passes in :pass array. XY [this box], [this board], [this set], and [pass] This pass is "double", and when there is less pass than 2, or when pass is not prepared, this group cannot connect this network. This group is made unusable and it progresses to the following group. When it is not the above: Save a total as a X-Y pass count of this board. L chips each count the number of the free passes in :pass array. LX [this box], [this board], [this L chip], [this group], and [pass] -- : to which this pass is "double", and this group cannot connect this network when there is less pass than 2, or when pass is not prepared -- this group is made unusable and it progresses to the following group. In not being the above, it saves a total as a L-X pass count of this L chip. To the next L chip of this board To the next board of this box the next box -- the following group -- stage 3C: -- selection [of a network]: -- each class which can connect a network -- : -- the minimum pass count is found out out of all the pass counts of this group. To the following group The greatest thing is found out out of these minimum pass counts. All the groups that have a pass count fewer than the greatest minimum pass count are removed after examination. This network cannot be interconnected when a group does not exist. When only a lot remains, the group of this network is chosen. When two or more groups remain, the following largest min is found out out of all the minimum pass counts. All the groups of a pass count smaller than this are removed after examination. This is repeated until a lot remains, or until the pass count of all the groups that remain becomes the same. One of the arbitration of the groups in which this network remains is chosen. Storage of all the pass counts of all groups is canceled.
stage 3D: -- in order to obtain or secure the convention:driver (or receiver) pass of procedure used below with the netlist of interconnect -- :1 -- pass is chosen from the chip number of the free element in the pass table of this level, and this node, and the chip number of a manager of a youth hostel node.
2) Mark the table element of the used pass.
3) Memorize which pass was used as a bus number in the driver (or receiver) pass number entry of this node. In order to derive an I/O pin number, the identity of the chip of this node and the chip (in or the case chip of a parent node) of a tea IRUDO node is checked from :12 piece the chip number and group number of a node. By this, the specific pass (for example, pass like L4-X5 or Board3-Y7) included is identified.
2) A pass number recalls that one pass in some pass which has connected the pair of a chip is shown. A chip, pass, and a pass number are given and the pin number which has connected this pass is read from the index table which has I/O pin number information. Using pass, in order to take out directions to a buffer (I/O), :1 pass number is obtained from this node. Or when tea IRUDOPASU is specified, a pass number is obtained from the tea IRUDO node. The driver or the receiver pass number was directed, and is made and obtained.
2) Obtain the I/O pin number of this buffer using a pass number.
3) Send out a base element statement to the netlist file of this node chip according to whether it is an input buffer or it is an output buffer. Under the present circumstances, while using an I/O network name so that it may be directed, the pin number obtained to that I/O pin is used.
The name of this network called netlist procedure: 'N' of interconnect is obtained. the [of each non-NULL level] -- 1L chip and after that, each node in this level [in / the whole :tree / on X chip, Y chip, and Z chip and] opens the netlist file about the chip of this node, when :preparation

of is not done. When level is X, Y, or Z: Each tea IRUDO node of the low order of this node sets :counter 'i' to zero. : (Childe is a driver) when it is D> 0 of Childe -- the case where it is D= 1 of this node -- :'N_D_c -- ' -- from -- directions are taken out to 'N' at an input buffer. ('c' is the number of a tea IRUDO node here.) The Childe driver pass is used in this case. When it is D> 1 of this node, directions are taken out from : 'N_D_c' to 'N_OR_i' to an input buffer. Under the present circumstances, Childe's dry pass is used and 'i' is incremented. When it is R> 0 of Childe: (Childe is a receiver) Take out directions from case [of D> 0 of this node, and R= 0 of this node]: 'N' to 'N_R_c' at an output buffer. Under the present circumstances, the receiver pass of an infant seat is used. When that is not right, directions are taken out from : 'N_R' to 'N_R_c' at an output buffer. Under the present circumstances, Childe's receiver pass is used. To the following tea IRUDO node : which takes out directions to :(node has the addition gate) 'i' input gate when it is D> 1 of this node -- it is NOR, when this network is a 'floating high' and the 'top sum' flag of this node is truth. It is OR when that is not right. Under the present circumstances, it has the input terminal connected to '(as opposed to all j of zero to i-1) N_OR_j', and the output terminal connected to 'N'. When it is D> 0 of this node and this node has non NULL parents: (a node is a driver) While securing driver pass, receive. Directions are taken out from 'N' to 'N_D' at an output buffer. Under the present circumstances, driver pass is used. When it is R> 0 of this node: (a node is a receiver) While obtaining and securing receiver pass, receive. Directions are taken out from 'N_P' to 'N_R' to an input buffer. Under the present circumstances, reception pass is used. To the next node of this level To the following level To the next cut network All opening netlist files are closed.

[0083] 2.4.7 Network drawing 47 [of two examples] a shows the original input design section of a simple network called 'BX'. This was equipped with one driver and three receivers, and has attained to two logic chips on one board in the same box, and one logic chip on the board of another side. The interconnect tree constituted by stage 3a of this network is shown in drawing 47 b. It must be cautious of how each logic chip, the node to each board, and the node to a box are prepared. The logic tip node supports the specific logic chip. The board node supports X chip on each board, and the box node supports Y chip. Z chip is unnecessary in this network. It is dependent on which group Y chip [which X and] are used chooses correctly, and this is not shown in the tree. The value of D and R is shown for every node. Even if a node is a receiver, it is cautious of L0 being D= 0. A node is a source node of this network and that reason is that it is not necessary to receive a value from a source node unlike other nodes. In the node of a board 2, initial value is 1 and the R count counts the receiver of L4. Since the source is a descendant, it is shown that R count was set to zero. The sent-out network name is shown using these networks. The structure of actual interconnect describes how the structure of a tree, D of each node, and R count are expressed. Drawing 48 a shows the original input design section of the tri-state network called 'EX'. This is equipped with six receivers which have attained to four L chips on three boards in three tri-state drivers which have attained to two logic chips on one board in the same box, and one logic chip on the board of another side, and two boxes. The interconnect tree constituted by stage 3a of this network is shown in drawing 48 b. Since this network has reached the box, it uses Z level crossbar. Since it has two tri-state drivers, it must be cautious of the node of a board 2 being D= 2. X chip is equipped with the addition gate and catches the term from L chip of a board 2. The same is said of the node of the box 2 which is the source of a network, and let this be "top sum." This Y chip is equipped with the top MOSUTO addition gate, and catches the term from boards 2 and 3. The node and its Z manager of a youth hostel node of a box 2 are equipped with the source, and make these R counts zero. It is indicated to drawing 49 how these are interconnected to be the actual gate and the actual buffer which are sent out to the netlist file about each logic chip and a crossbar chip. It must be careful of how each tri-state driver is changed into the AND gate with layout conversion. These outputs are caught by X and the addition gate of Y level. A reception input is transmitted from Y chip of the "top sum" node 2, i.e., a box. The receiver of a box 2 drives with the pass connected to interconnect. The receiver of a box 6 drives through Z level crossbar chip.

[0084] 3 Application 3.1 of a rear riser system A rear riser logic simulation system logic simulator is a system realized by hardware or software. This system outputs a stimulus of a lot while

receiving the direction of the stimulus to an input design and the layout section of a lot, and the stimulus during a certain period. It predicts realizing the actual input design section and generating the same predetermined stimulus by this stimulus. A stimulus and a response transmit the logic condition of a specific layout network to specific time amount. It is an important property that a simulator user supplies only description of the layout section with the gestalt of an input design file, and while changing the layout section for a short period of time, a stimulus can be again given to this. The operation of the current SOSUTO wear logic simulator layout section uses the computer software program, and performs the sequential algorithm which predicts the operation of the layout section ("An Introduction to Degital Simulation", Mentor Graphics Corp., Beaverton, Oregon, 1989). Either the code algorithm by which the event drive was carried out, or the compiled code algorithm is used as known well. It is constituting the hardware which performs the code algorithm as being used for a software simulator with the same operation of the current hardware logic simulator layout section by which the event drive was carried out, or the compiled code sequential algorithm. By realizing development and/or special algorithm operation for the parallel processing of an algorithm directly, hardware can obtain the profits by the activation. This is impossible in the computer activation software of the general purpose. A current hardware logic simulator operates by performing the sequential algorithm which predicts the response of the input design section. A new means to constitute a logic simulator is based on the rear riser system. A rear riser logic simulator system receives an input design, and changes this into the logic of rear riser hardware, and the configuration of an interconnect chip. Under the present circumstances, rear riser layout conversion system is used. A rear riser logic simulator system receives the stimulus to the layout section of a lot, and the direction which a certain period simulates, gives a stimulus to the layout section realized through vector memory, and catches the response of the lot from the layout section realized through vector memory. The response supports generating the same predetermined stimulus by actually realizing the input design section. The reason is that it is made to correspond to said stimulus and hardware actually realizes the layout section. A rear riser logic simulator realizes the actual layout section, and this differs from all the present logic simulation systems in the point of opting for the response to a stimulus of the layout section, although all the current logic simulation systems perform the sequential algorithm which predicts the response to a stimulus of the layout section. The main advantages are quick rather than a sequential algorithm can predict a response, with various speed, I hear that the realized layout section generates a response, and it has them. The rear riser logic simulation system consists of rear (it already explained) riser layout conversion system, a logic simulator stimulus and a response transmission system, a rear riser hardware system and a host computer, and the logic simulator that operates a kernel conjointly (drawing 50).

[0085] 3.1.1 While changing the system of ********** of a logic simulation stimulus and a response into the binary file containing the stimulus data which can load the stimulus event input file which the user created to direct vector memory, a user changes a response into the response event output file in which read-out is possible from the file which has binary response data read from vector memory. The stimulus and the response event consist of a network name, time amount, and the status value of a new network. Conversion is the conversion between a network name and a vector memory bit, and conversion between the 'real time' of a simulation, and a vector memory location. Time amount conversion is reported as what generated the response event in this vector memory location at this time of day while inscribing each specific time amount which has a stimulus event on a vector memory location. In the suitable example, the stimulus input event file and the responded-output event file are made into Mentor Graphics Logfiles ("Quick Sim Family Reference Manual", MentorGraphics Corp., Beaverton, Oregon, 1989). This is a text file including a series of time of day, network names, and new network status value. With the batch simulation interface tool in an EDA system, while creating a stimulus input event file, a responded-output event file is interpreted. At a suitable example, it is Mentor Graphics about this tool. It considers as a RSIM tool. Here, all base elements are assumed to be what is simulated for zero delay so that this section may explain later. In order to change a stimulus event input file into a stimulus binary file, :1 stimulus input event file is read. While

carrying out order of the stimulus event according to the passage of time, it is determined how many different time of day have an event.
2) Read the correspondence table to each vector memory in this layout section that layout conversion system outputs.
3) Each vector memory location supports the time of day which has one or more stimulus events. When sufficient vector memory location does not exist in each different stimulus event time of day, steps 5 and 6 are repeated as required, and sufficient stimulus binary file for such all time of day is outputted to it. This file has the stimulus which suits memory respectively.
4) Assign storage of a vector array "V0", "V1", etc., etc. The each is in agreement with the number of a location, and network width of face, and the vector memory used for the layout section simulated is used. Storage of the time array "T" which has the same length as a vector array is assigned. "Last vector" A buffer "B0", "B1", etc. are assigned. To each vector memory, this buffer is the same width of face as that network respectively, and initializes these to zero.
5) Set a vector array index counter 'v' to zero. In the 1st earliest time of day, vector memory 'n' and vector memory bit BOJISHON 'i' of this network are set up among each time of day which has one or more stimulus events. Under the present circumstances, the correspondence table entry of the network of this event is used. About the new value over this event, it is Vn [v]. It writes in Bit i and the Bn bit i. To the following event V0 [v] and V1 [v] etc. -- each of the contents is written in B0 and B1 grade. This time of day in T [v] is memorized. v is incremented. To the next time of day which has a stimulus event
6) Write the vector array V0, V1 grade, the time array T, and a cycle count 'v' in a stimulus binary file. In order to change a response binary file into a response event output file, :1 vector array V0, V1 grade, the time array T, and a cycle count 'v' are read from a response binary file. Each vector memory location is in agreement with the time of day which has one or more stimulus events. To respectively different stimulus event time of day, when a vector memory location is not enough, a repeat and all response binary files are read into these arrays as required in steps 1-4.
2) Read the correspondence table to each vector memory in this layout section which layout section conversion system outputs.
3) A "last vector" buffer "B0", "B1", etc. are assigned. To each vector memory, this has the same width of face as the network respectively, and initializes this to zero.
4) Set a vector array index counter 'v' to zero. Each location in a vector array: Compare V1 [v] with B1 for V0 [v] as compared with B0. The differences of the bit of Vn [v], and Bn: Arrange the network name corresponding to the vector memory and the vector memory bit position of this bit. Under the present circumstances, the correspondence table to this memory is used. A new response event is written in an output file. Under the present circumstances, the value of a network name and a new bit and time-of-day T [v] are used. Each of the contents, such as V0 [v] and V1 [v], is written in the following event at B0 and B1 grade. v is incremented. To the next location
[0086] 3.1.2 A logic simulation operating kernel operating kernel constitutes the rear riser system of the layout section simulated, and it catches a response while it gives a stimulus. A host computer performs this. It controls a clock generation machine and a reset generator through a host interface while an operating kernel constitutes a logic chip and an interconnect chip, and reads vector memory and layout memory and writes them in, as explained to each section. In order to perform a simulation, while reading the configuration file of :1 layout section, as the section of a configuration explained, all rear riser logic chips and interconnect chips are constituted using this. While reading preliminary-design memory data from a file, this is written in layout section memory.
2) Read a stimulus binary file. The contents of the vector array are memorized in corresponding vector memory through a host interface.
3) Clear the contents of all the vector memory in a vector memory module. A layout section reset generator is period-ized and the layout section realized is initialized.
4) The clock generation machine of the ECLK network of the 'v' period is enabled. By this, vector memory can send out these stimulus data, and while operating the layout section realized

according to this stimulus, vector memory catches response data. This is explained in the section of a stimulus/response.
5) While reading the contents of vector memory, memorize these to a response binary file with a time array "T" and a cycle count "v."
6) Since the capacity of vector memory is inadequate, when preparing many stimulus binary files from 1, repeat steps 2-5 for every file.
7) Save the layout memory content in the file for a user trial.
[0087] 3.1.3 Using the use rear riser logic simulator of a rear riser logic simulation system, in order to simulate the input design section, by marking the network for catching the network which should be stimulated, and this response using the property which shows :1 vector memory connection, use the layout creation tool of an EDA system and prepare the input design section. If required, a preliminary-design section memory data file will be prepared. A stimulus event input file is prepared using the batch simulation interface tool of an EDA system.
2) Change the input design section using rear riser layout conversion system, and output a configuration file and the table file corresponding to a vector memory circuit network.
3) Make a stimulus and response conversion system run, and change a stimulus event input file into a stimulus binary file.
4) While making an operating kernel run and performing a simulation, output a response binary file.
5) Make a stimulus and response conversion system run, and change a response binary file into a response event output file.
6) Translate a response event output file using the batch simulation interface tool of an EDA system.
7) The input design section, a preliminary-design section memory file, and/or a stimulus event input file are changed so that it may be shown by the result of a simulation, and if required, steps 2-6 will be repeated. In the modification of the conversational mode of a rear riser logic simulation system, the sampler is used to the response using SUTIMYURETA to the stimulus. A configuration and operation are the same except for the following. That is, a stimulus and response conversion system are communicating with the direct operating kernel using the conversational mode SHIMYURESHONINTAFE stool operating kernel which uses an interactive-simulation interface tool instead of a batch simulation interface tool, is carrying out direct communication to a stimulus and response conversion system instead of minding a file, and operates to coincidence instead of minding a file. Each time step which has an event is created for the edge detection type simulator instead of a vector memory location.
[0088] 3.1.4 In the implementation rear riser system of three or more logic states, two logic conditions are realized, a low logic condition (L), i.e., a false, practical :high logic condition (H), i.e., truth, which is things, and it realizes by realizing each network directly in the input design section using the single signal of a rear riser system. In logic simulation environment, to express the condition of three or more logic signals occasionally is desired. For example, the 3rd logic variable or ambiguous logic condition which is not initialized using a condition and "strange (X)" is expressed. A high impedance condition (Z) is useful to realizing the bus of wire association like a tri-state bus. In some examples of a rear riser system, a high impedance condition is directly realizable. For example, the layout section is realized by the tri-state bus of a rear riser system as long as it has the capacity for a logic chip and required interconnect of arbitration to constitute a tri-state bus function, when a tri-state network is needed for the layout section. The number in the condition that the logic condition of arbitration is instead realized by encoding one network as follows to two or more signals and that it should carry out :implementation is determined. The minimum binary number of bits needed for encoding all conditions individually is determined, and this is called 'n'. The actual binary logic signal of the 'n' individual realizes the network in the layout section. For example, when you need three conditions (H, L, X), it realizes the single layout network in a rear riser system using two actual binary signals. This conversion is performed between base element conversion stages, these new binary signals are inputted into a layout section data structure, and the Original Engineering Consult. network is replaced. Furthermore, the logical-circuit network which operate according to a condition logic function

realizes the logic base element in the layout section. For example, when using three conditions (an H= high logic condition, an L= low logic condition, X= strangeness), 2 input AND gate in the layout section is realized according to the logical circuit which operates according to a tri-state AND function (drawing 51 a). When setting one of input terminals to X and setting for an input terminal not to exist to L, a logic function operates like a tri-state simulator using X condition produced in an output terminal (drawing 51 b). This network has two 2-bit input terminals and one 2-bit output terminal (drawing 51 c). This technology of realizing a multi-state can be used for a part of whole input design section or layout section so that it may be needed for layout analysis. The network simulated in the state of [pieces / two] many is carried out in this way, it creates in an input design file, and a layout section reader creates the network of a large number to an above-mentioned alternate circuit network and an above-mentioned base element paying attention to this in a layout section data structure. [as opposed to a base element in a base element converter] When the logic base element has mixing with network connection of two conditions, and the network of three or more conditions, it uses the logical-circuit network which operates according to the conditions of a network. If that is not right, it will become the simulation actuation mentioned above.

[0089] 3.1.5 Make the time delay at the time of a signal passing a logic element from the expression current logic simulator of the delay of a rear riser by various methods. Since the logic in the logic chip of a rear riser is actual hardware, the delay property cannot be specified correctly completely and cannot create logic delay directly. the special method in the simulator which performs a program is used for logic delay -- and/or, it creates between layout translation processes by inserting the special logic function which forms delay. Delay can be formed as zero delay, unit delay, or real delay during the simulation to realize. This selection is accomplished by the user and specifies this as a rear riser logic simulator system.

[0090] 3.1.5.1 Without forming real-time delay, zero delay zero delay processes delay as zero, and means performing a simulation. For example, when a stimulus event arises in the input terminal connected with the output terminal only through joint logic at time of day 'i', the response event of this output terminal is reported as what is produced at time of day 't'. For zero delay, layout section conversion system does not insert a special logic function. Like the above, a simulation is performed according to the method explained in the MEINRIA riser logic simulation system.

[0091] 3.1.5.2 It will become complicated if the delay dependence function of arbitration is prepared in the delay dependence functional-design section. It does not result even here in a zero delay timing model. Like a closed loop function, i.e., the crossing KAPURUDO gate, when preparing asynchronous feedback, storage is formed unconditionally. It depends for the memory storage function on relative delay. A delay dependence function will become the thing of other gestalten, if delay is used for an open loop function. There is the IKUSUKURUSHIBU OR gate which has the delay element connected with the input terminal as this example (drawing 52 a). The output of the IKUSUKURUSHIBU OR gate is a high in the time amount needed for a signal spreading through a delay element. Change of the signal supplied to this network outputs a pulse to an output terminal (drawing 52 b). Since actual rear riser logic delay is not zero, when the direct control of this cannot be carried out, in the case of many closed loops and some open loops like the crossing KAPURUDO gate, a delay dependence function operates correctly. However, a user demands to operate so that the layout section realized certainly may mean. A current timing tool for analysis detects automatically a header and the action for which it depends on open loop delay while reporting for the moment of asynchronous feedback. If a user needs rear riser layout conversion system, it will perform timing analysis by using a timing tool for analysis. In the example of suitable implementation, a Mentor Graphics Quick Path timing tool for analysis is used ("Quick Path User's Manual", Mentor Graphics Corp., Beaverton, Oregon, 1989).

1) As a part of layout translation process, an ERCGA netlist conversion tool outputs evaluation of internal interconnect and logic delay. These are sent out to a report file.

2) Input data into read-out and a layout section data structure from a report file after changing all netlists. Under the present circumstances, each delay evaluation relevant to a base element

or a network is used.
3) Write a layout section data structure in a layout file.
4) Apply a timing tool for analysis to a layout file. Change to which the arbitration detected by the timing analyzer may happen is reported to a user. A user evaluates an input design file appropriately and corrects it.
[0092] 3.1.5.3 With unit delay unit DIREIMODERU, form so that each logic base element may have the delay of one unit (unit). Using a delay dependency, such formation is sometimes used for the layout section, and right actuation is guaranteed. By applying a suitable property to the base element in an input design file, a user specifies a zero delay base element and the compounded unit delay base element. Automatically, a flip-flop is formed in the output terminal of each unit delay logic element, and unit delay is formed. These flip-flops are connected to a common clock, and this common clock accomplishes 1 time of a period for every unit time amount of a simulation with the 2nd clock generation vessel. These flip-flops and a 'timer clock' network are added to design-data structure according to a base element translation process. There is a flip-flop created using the crossing KAPURUDO gate as an example of the logic layout network simulated using unit delay (drawing 53 a). In the output terminal, a unit delay flip-flop is formed and each gate is constituted (drawing 53 b). The final operation which gives a continuous timer clock and a continuous input signal is the operation of a flip-flop of having the unit delay gate (drawing 53 c). Although the rear riser logic simulator for a unit delay simulation is accompanied by the following modification, it operates using the same method as zero delay.
- A user directs whether the time amount of which about is equivalent to one unit.
- Restrict a stimulus and the count of a response to the multiple 'M' of the time basis specified by a user.
- Each vector memory location is unrelated to whether support M time basis and the stimulus event at this time exists.
- A stimulus and response conversion system create a map between an event and a vector memory location according to these correspondence relation using such specifications.
- Finally the time of day which does not have the stimulus event is expressed by the vector memory location which has the same contents as a pre- location.
- An operating kernel sets the frequency of a 'timer clock' clock generation machine by M times the frequency of ECLK, and directs to operate, while taking a synchronization mutually. A stimulus and response of one ECLK, therefore a lot exist every M time bases between operation.

[0093] 3.1.5.4 Realize real delay real delay, i.e., the delay by various time bases, using the specific hardware configuration in a logic chip. This hardware configuration is automatically inserted in the layout section data structure of each real delay logic element during layout conversion. A serial shift register is constituted in :each logic base element output terminal and serial which have some technology in this. The number of the delay units for which it is needed at each case is made to correspond, and the length is constituted. The clock of all the shift registers is carried out in common 'a timer clock' so that 1 time of a period may be made to each time basis. Thus, a shift register acts as 'n' unit (unit) real delay. 'n' is the length of a register here (according to the value in drawing 54 a and a delay register, chosen through a multiplexer). In instead of, a finite-state machine (FSM) and the counter which has the storage to one or more starting counts (starting count) are constituted at each logic base element output terminal and a serial (drawing 54 b). FSM detects conversion of a logic base element output state. In conversion of each condition, a counter is loaded by FSM using a suitable starting count for the state transformation (a rise (rising) or descent (falling)) of the generated specific class. All counters are period-ized in common 'a timer clock' so that 1 time of a period may be accomplished for every time basis. When a count becomes zero, FSM spreads output state conversion to the output by which delay was carried out to delivery and its connected input terminal (refer to drawing 54 c).
[0094] 3.1.6 The transmission rear riser logic simulator system of the condition to other simulators [simulator / rear riser] has the advantage of being very high-speed, and, for this reason, can process it by many various test cycles rather than software or the simulator by

other event drives. This system cannot display delay and the item relevant to other time amount, and also has the disadvantageous point that no nodes in the layout section can be supervised. Although the software simulator by the event drive of common use is quite a low speed, it has the advantage that access to all the network nodes for an expression, stimulus, and monitor of an item can be performed. However, since the software simulator by the event drive of common use is very a low speed, sending to the condition of having made the mistake in what 1 million or what 1 billion periods separating does not produce the simulated layout section from an initial state in fact. It turns out that the condition of having mistaken cannot actually happen. When it constitutes a rear riser system using the logic (it is (like Xilinx LCA)) chip which can read the value of an initial state, i.e., an internal flip-flop, and a logic gate output, the simulation realized is stopped and the condition of the whole layout section is read. By combining a rear riser logic simulator and other simulators, the condition (namely, value of all the internal storage in the layout section) of the simulated layout section is transmitted to another side from one side. :1 [under the present circumstances,] according to the following methods -- the same layout section is loaded to both simulators.
2) The layout section in a rear riser logic simulator is simulated as follows between number cycles. That is, the layout section is changed into the condition before the error which should be supervised by details, or the occurrence of other conditions for a short time.
3) At this time, a rear riser stimulus clock is stopped and all the conditions of the layout section are read from a logic chip.
4) Initialize the layout section currently expressed by other simulators, and make the condition of being read from the simulator based on a rear riser suit at this time.
5) Advance a simulation to other simulators.
Thus, using an ultimate speed of a rear riser logic simulator, since it is too long, the error which cannot be removed by other methods can be removed and the cause of an error can be analyzed using other details and visibility of a simulator.
[0095] 3.2 Rear riser fault simulation system fault simulation is the layout section and the group of the stimulus used for generally testing the accuracy of an integrated circuit after deformation of the logic simulation used for developing and correcting a test vector, i.e., an assembly. While simulating the fall tee version (faulyt version) by user layout using a test vector stimulus, as compared with a good (good) version, a test vector stimulus investigates whether response with the another response of a good version is generated. If another response is generated, it is shown that the test vector stimulus has detected failure (fault). This is repeated to a set with much failure. This aims at developing the test vector of the lot which detects as many failures as possible. Generally, two failures are simulated in each network of the input design section. That is, there are a case which a network calls "stack ATTORO (stack-at-low)" where it is always a low state, and a case called "stack ATTOHAI (stack-at-high)" where it is always a high state. general -- the input design section -- thousands of networks and a test vector -- having -- and fault simulation -- each -- ****** -- since it is repeated for every version of a test vector, this is a task which starts extremely as for time amount. A new means to constitute fault simulation is based on the rear riser system. The method of a rear riser logic simulator is used about correction of fault simulation. : using serial fault simulation technology ("Quick Sim Family Reference Manual", Mentor Graphics Corp., Beaverton, Oregon, 1989) -- the layout section by which :1 implementation was carried out about each failure is corrected, and failure is told.
2) Give a stimulus, operate the layout section and sign difference with a flag in a response as compared with the response of good layout.
3) Remove failure and record whether the difference by this failure exists.
To performing the sequential algorithm with which the present fault simulation system predicts the response to a stimulus of the failure layout section, by rear riser fault simulation, the actual failure layout section is realized and both are different in the point of opting for the response to a stimulus of the layout section. The main advantages are emitting a response at the rate of quick versatility rather than it can answer an algorithm with the sequential layout section realized. Failure is told to the direct layout section as constituted from rear riser logic and an interconnect chip. In order to tell failure to an input-design network, when the network of :input-

design section has the correspondence network in the logic chip: Reconfigurate each logic chip connected to the network using a fault configuration. This is the same as that of an original configuration except for the point of having connected to fixed yes or a fixed low the input terminal connected to the network according to failure. Inclusion of the network where the network in the input design section does not have the network [be / it / under / logic chip / correspondence] and which case [the network] : corresponds is carried out to the logic function of a logic chip, and it reconfigurates a logic chip using a fault configuration. This is the same as that of an original configuration except for the point which carries out the logic function by which inclusion is carried out to a network to the configuration which operates so that a network may always serve as yes or a low according to failure. In order to remove failure, a chip is reconfigurated using an original configuration. Although a rear riser fault simulator has the following differences, a rear riser logic simulator and :rear riser fault simulator which is the same as that of an essential target (drawing 55) are fault configurators (configurator), and it constitutes the additional portion of the layout section conversion system of the high order of a logic simulator. :1 to which a rear riser fault simulator outputs the difference of a configuration file to each following failures -- failure is temporarily told to a layout section data structure.
2) Determine which logic chip is influenced by change of the layout section by failure.
3) Send out the netlist file of a carrier beam logic chip for effect.
4) Output the configuration file of a carrier beam logic chip for effect using an ERCGA netlist conversion tool.
5) Compare a fault configuration file as it is original, and save only a difference to a configuration difference file.
Instead of constituting response vector memory to a response network, a layout section converter constitutes fault response memory. As the section of a stimulus/response explained, these fault response memory sets a flip-flop, when difference is detected in a response network as compared with the good value memorized in memory. An operating kernel acts variously to fault simulation. In order to operate fault simulation (zero delay is shown.) the same is said of a unit or real delay -- using read-out and this, :1 layout section configuration file is constituted for all rear riser logic and interconnect chips, as the section of a configuration (configulation) explained. Preliminary-design section memory data is written in read-out from a file, and this is written in layout section memory. A configuration difference file is read.
2) Read a stimulus binary file. The contents of the vector array are memorized in the stimulus vector memory which corresponds through a host interface. A time array "T" and a cycle count "v" are read. The response binary file of a good circuit is read. The contents of the vector array in corresponding fault response vector memory are memorized.
3) Output the fault configuration file of the logic chip influenced by 1st failure using a difference of the configuration of this failure. Moreover, the logic chip to this failure is constituted using these.
4) Clear all the vector memory counters and difference detection flip-flops in a vector memory module. A layout section reset generator is period-ized and the layout section realized is initialized.
5) The clock generation machine of the ECLK network of the 'v' period is enabled. By this, stimulus vector memory sends out these stimulus data, and while being able to operate the layout section realized according to a stimulus, fault response vector memory compares response data to a good circuit.
6) While checking a fault response detection flip-flop, record whether the difference arose to this failure.
7) Return an original configuration to the broken logic chip.
8) Repeat steps 3-7 to each remaining failure.
[0096] 3.3 Many of present idiomatic simulators in a rear riser logic simulator evaluation-system current EDA system operate according to either the sequential algorithm which is called an event drive and which was known well or the compiled code simulation ("An Introducion to Digital Simulation", Mentor Graphics Corp., Beaverton, Oregon, 1989). In the 1st algorithm, each base element in the input design section "is evaluated" for every time step. In this case, the

network which drives the input pin of a base element has change of an event, i.e., a condition. Moreover, in the 2nd algorithm, each base element of the input design section is evaluated to all time steps. Evaluation of a base element is actuation as which an output value with a new base element determines what kind of thing it is to a new input value. This is repeatedly produced between simulations. Usually, 1 operation estimates only a small base element like the gate. Under the present circumstances, an index table or other direct technology are used. Generally a large-scale logical-circuit network is simulated as a combination of a small base element and a network. The internal evaluation which much time amount requires is needed for every evaluation of each large-scale network. It is in the exterior of a rear riser system, and the logic simulator which performs a sequential simulation algorithm is combined with a rear riser logic simulator evaluation system. This evaluates one or more large-scale logical-circuit networks under algorithm simulation using the hardware of a rear riser. A single base element expresses each large-scale logical-circuit network which should be evaluated by the rear riser system in an external logic simulator. One of the profits of these is in that speed. The reason is that the realized base element is evaluated almost momentarily. The size of the logical-circuit network evaluated by the rear riser system is restricted by only the logic capacity of a rear riser, and includes the logic capacity of all the input design sections and tales doses. The rear riser logic simulator evaluation system serves as conjointly a rear riser hardware system and a host computer from rear riser layout section conversion system (it already indicated) and RIARAIZA logic simulation IBARYUETA (evaluator) (drawing 56). . This is combined with the external logic simulator which operates a sequential simulation algorithm. In order to prepare the logical-circuit network for evaluation with a rear riser logic simulation evaluation system, the logical-circuit network which should be evaluated by the :1 rear riser system is assembled as the input design section of an EDA system.

2) Direct to drive a property with a nest, a simulator, and a sampler to the I/O network of each logical-circuit network, respectively.

3) Change the input design section using rear riser layout section conversion system by the usual method, and output a configuration and a correspondence table file to this aggregate of a logical-circuit network. In order to perform a simulation, according to the following methods, an external logic simulator is operated, and while performing a simulator algorithm, rear riser logic simulation evaluation equipment is also operated. : Constitute the data structure of 1 external simulator and prepare a single base element for each [which should be evaluated by the rear riser system] logical-circuit network of every.

2) Read the correspondence table file of the layout section and relate base element I/O with these addresses of a rear riser host interface bus.

3) As the section of a configuration described, read the configuration file of the layout section and constitute all rear riser logic and interconnect chips using this. Preliminary-design section memory data is read from a file, and this is written in layout section memory. A layout section reset generator is made to period-ize, and the logical-circuit network realized is initialized.

4) Initialize all simulators using initial value.

5) Operate the simulation algorithm of an external logic simulator. :1 which evaluates the base element based on a rear riser by the simulation algorithm using this method -- since this value is loaded while transmitting the value over all the inputs to this base element in this simulation time step to RIARAIZA logic simulation IBARYUETA, it sends to a simulator.

2) Point so that all the output samplers of this base element may be checked to RIARAIZA logic simulation IBARYUETA, and even if it is change to what kind of output, retransmit to a simulation algorithm.

6) In order that a user may make a trial and correction before and after a simulation, in order to access a layout section memory content through a host interface, give the function of the user interface system of an external logic simulator.

When performing a simulation algorithm in software, while performing this with a rear riser host computer, a simulator, a sampler, and layout section memory are accessed using a host interface. When performing a simulation algorithm by hardware, a simulator sampler and layout section memory are accessed using the communication link to a host computer. The direct

continuation between simulator hardware and the device (USD) module by user assignment of a rear riser is used for modification of a hardware simulator system. This method connects these to the USD base element corresponding to the evaluation unit of a hardware simulator instead of directing the simulator and sampler about I/O of a base element of the same :1 input-design section as the above, although accompanied by the following differences.
2) Connect the evaluation unit of a hardware simulator to USDM of a rear riser electrically. If an input event occurs, while supplying a new value to the base element realized by direct continuation, an output response is caught by direct continuation without a host. For this reason, a remarkable high-speed evaluation speed is obtained.
[0097] 3.4 When realizing the rear riser prototyping system input layout section, it can realize as a prototype of the direct layout section, and this can be operated. Generally the timing delay of a rear riser system is not in agreement with the timing delay by implementation of ultimate hardware, for this reason, a prototype cannot operate at full layout speed, but the layout section can almost operate actually in the real time with the prototype of the rear riser base. The layout section realized is simulated in a rear riser clock generation machine, the simulator controlled through a host, the hardware device which a user actually specifies, and the virtual meter (it explains below) realized, and/or a self-simulation is carried out according to internal logic and/or a layout section memory content. It analyzes, while carrying out the monitor of the operation of the layout section through a host, the hardware device which a user actually specifies, and the virtual meter realized using the sampler controlled by investigating a layout section memory content. A designer has a dialog with the layout section in the real time like "bench top (benchtop)" environment directly. The rear riser prototyping system is equipped with layout section conversion system and a prototyping system with the rear riser hardware system and the host computer (drawing 57). A prototyping operator constitutes the rear riser system of the layout section which operates, and supports an interactive stimulus and response of the rear riser layout section. This operator performs in a host computer and answers to a user's command through the control program which runs in direct or a host computer. In order to operate the layout section realized, as the section of :1 configuration described, the configuration file of the layout section is read and all rear riser logic and interconnect chips are constituted using this. Preliminary-design section memory data is read from the file which a user supplies, and this is written in layout section memory. While reading a corresponding table file, correspondence, and SUTIMYURETA, the samplers and these host interface bus addresses between layout section network names are established.
2) Period-ize a layout section reset generator and initialize the layout section realized.
3) Process :-user command which performs the following actuation if needed continuously, and control a clock and a reset generator.
- Process a user command and change a SUTIMYURETA output value. Under the present circumstances, the network name which a user gives is related with corresponding SUTIMYURETA using a correspondence table.
- Process a user command and display the data input value of a sampler. Under the present circumstances, the network name which a user gives using a correspondence table is related with a corresponding sampler.
- Process a user command, and while reading the location in a layout section memory module, write in. It checks that the layout section is not operating. Under the present circumstances, before accessing layout section memory, a clock generation machine stops, and it is confirmed that actuation of unsuitable layout section memory is avoided. It reports to a user whether the layout section is stopped. In order to use a rear riser prototyping system, :1 input-design section is created in a host EDA system.
2) Mark the layout section network which should be connected to a simulator, a sampler, and a clock or a reset generator.
3) Prepare a layout section base element, a network, and connection, and design the network to the virtual meter of arbitration which should be used (refer to following).
4) Change the input design section using rear riser layout section conversion system, and output the configuration file of the layout section.

5) Operate the layout section using a rear riser prototyping operator. In the specific example shown by drawing 57 , the digital computer layout section is realized using a rear riser prototyping system. A user expresses the computer logic in an input design file, and the layout section of memory using a host EDA system, and a user changes into a configuration file using rear riser layout section conversion system. In an actual example, the front panel control input and display output which are connected to an actual front panel control switch and an actual indicator are specified in the input design section, and are connected to SUTIMYURETA and a sampler under the user control through a prototype operator. It is specified that the clock input signal of a computer is outputted with a rear riser clock generation vessel. In order to operate a prototype computer, a user makes a rear riser prototype operator run, and constitutes a rear riser system according to a computer-aided design. While loading a computer program code so that activation may become possible in the computer-aided design section realized, the initial data is loaded to layout section memory through a prototype operator at the time of initiation of operation. If a user makes a clock generation machine enable, the computer-aided design section actually operates in the logic and the interconnect chip with which rear riser hardware was constituted, and it will read and write in the data in layout section memory while it performs the program instruction code read from layout section memory. A user operates a front panel control-input terminal, and reads a working display output through a prototype operator's correspondence SUTIMYURETA, and access to a sampler. Reading appearance of the result is done by the user out of memory through a prototype operator at the time of termination of a program. A user analyzes this result and judges whether according to an intention of whether the layout section being exact and a user, it is operating correctly. When the layout section is not operating correctly for the layout error in the input design section, a user corrects the **** error for host EDA systems, and repeats a prototyping process.

[0098] 3.4.1 When the virtual meter stimulus and/or analyzer machine which are realized are needed in a prototype debugging process, direct continuation is carried out to the layout section realized through the device module with which a user gives the meter of common use like a logic analyzer. In order to connect an actual meter, it connects with the layout network which should be connected to a meter, and while preparing the base element which shows the meter USD in the input design section, the USD use file which has specified ESD connection is created. A meter is connected to direct USDM, the layout section realized as mentioned above is transformed, and it is made to operate at this time. Furthermore, the base element and network which form a "virtual meter" in the layout section in an input design file, and are realized using this layout section are prepared. For example, when a logic analyzer is used as the meter which carries out the monitor of the logic signal of a lot and which was known well and these satisfy certain trigger conditions, while sampling-izing continuously the signal with which the lot was analyzed, these values are recorded into memory. Reading appearance of this is carried out after that for analysis. Drawing 59 showed the configuration of a virtual logic analyzer, and this analyzer is equipped with response vector memory, the condition detector which has a logic base element, one or more SUTIMYURETA and a sampler, and other logic base elements. In addition to :1 layout section for using the layout section, and realizing and using a virtual logic analyzer, as shown by a diagram, the base element to these components in the input design file which interconnected is prepared. Especially, a response vector memory input is connected to the layout section network which should be analyzed, a condition detector input terminal is connected to the layout section network by which a monitor should be carried out on trigger conditions, and the logic of a condition detector is specified by this according to the conditions which should be detected.

2) Change an input design file into a configuration file according to the usual procedure.

3) Constitute the layout section in a rear riser prototyping system.

4) Make a "reset" signal period-ize through a simulator, and give the stimulus needed for the layout section realized starting actuation.

5) The monitor of the sampler "by which the trigger was carried out" is carried out. When it is shown that the signal "with which the trigger of the sampler was carried out" is truth, a logic analyzer catches the analyzed signal data.

6) Read this data from the response vector memory of a logic analyzer through a host interface. It analyzes, while displaying this using the same thing as a general computer debugger program or this general. This is an example which shows how a virtual stimulus or an analyzer machine is realized using the layout section in a rear riser system. A concept of the meter itself like the concept of a logic analyzer is cautious of it not being new. It is an element of freshness to realize a meter using the input design section in a rear riser system.
[0099] 3.5 Using a rear riser executive system rear riser executive system, it is specified in an input design file, and or it is not yet constituted, perform the hardware function which does not try to never constitute in eternal hardware. While constituting :eternal hardware from which some advantages are acquired by performing this, the layout section realized for the purpose of a software development or others is used. When becoming possible to carry out while creating to a software development, debugging this and not using eternal hardware by this, it prepares so that software can be used. A rear riser executive system is used in order to play a role of a universal hardware device and to perform various transposition needed. When a special function is required, with a host computer, the configuration file of a hardware system and other files are called from storage, constitute a rear riser system according to this layout, and perform a function (when rear riser layout conversion system is realized). For example, in electric layout environment, the role of the logic simulation hardware accelator needed, routing hardware accelator, or a hardware graphics processor is played using a rear riser executive system. In the environment where a digital signal is processed, the role of the synthesizer which has the real-time spectrum analyzer or the special effect needed is played using a rear riser executive system. A rear riser executive system considers that an input design is a right thing except for the following points not using the meter for the :1 analysis which is the same as that of a rear riser prototyping system. SUTIMYURETA, a sampler, and the role performed only using layout memory access are controlled, and data is outputted and inputted.
2) The controller which directs the function in which specification is performed can be created, and give the function to control and perform a rear riser prototyping operator using this to the input terminal / output terminal, and the control interface suitable for use of a function.
[0100] 3.6 Create automatically the example of implementation which cannot be reconfigurated [that the input design section is eternal and] using the modification of rear riser production-system rear riser layout conversion system. The rear riser logic chip of the same class as this eternal example of implementation is consisted of by the layout section realized and a number is used. In a rear riser production system, while constituting the logic device equivalent to an ERCGA logic chip which it is eternal and cannot be reconfigurated in a function using the ERCGA netlist conversion tool, automatic printed circuit board (PCB) arrangement and a routing tool are driven ("Getting Started with Board Station", "Layout User's Manual", Mentor Graphics Corp., Beaverton, Oregon.1989). Under the present circumstances, PCB which interconnects eternally the logic device which cannot be these-reconfigurated is manufactured using the specification about logic chip interconnect. In the suitable example, LCA is used as an ERCGA logic chip. By manufacturing LCA, the logic chip equivalent to LCA which cannot be reconfigurated is functionally offered with the gestalt of the LCA chip combined with the configuration PROM memory ("The Programmable Gate Array Data Book", Xilinx, Inc., San Jose, 1989). The binary file used for programming PROM by the LCA netlist conversion tool is created. Moreover, LCA is equipped with logic, and in case LCA supplies power using this, it can constitute the LCA itself. Under the present circumstances, this will be used if there is a PROM. The rear riser production system is equipped with party SHONA used for the ERCGA netlist conversion tool which is deformation although used into the same layout section reader as having mentioned above, a base element converter, rear riser layout section conversion system (RDCS), interconnect and a network listing system, and RDCS, automatic PCB arrangement, and a routing tool (drawing 60). The rear riser production system is not equipped with the rear riser hardware system or the host computer. This reads an input design section file and PCB specfile. While reading an input design section file using :1 layout section reader which operates according to the following methods, a layout section data structure is created.
2) Change a layout section data structure into a logic chip base element using a base element

converter.
3) Assign a base element to a specific logic chip using party SHONA.
4) Create the netlist file for a logic chip using interconnect and a network listing system. Instead of supplying the netlist file for an interconnect chip, a single interconnect file is sent out with the gestalt accepted in automatic PCB arrangement and a routing tool in the list of cut networks and these logic chip I/O pin connections.
5) Supply a binary configuration file for every logic chip with the gestalt suitable for the configuration of the equivalence logic device which cannot be reconfigurated using an ERCGA netlist conversion tool.
6) Using automatic PCB arrangement and a routing tool, read into an interconnect file and PCB specfile (this file is equipped with the physical information which is not directly related to logic layout of the size of PCB, a connector requirement, etc.), and output a PCB manufacture data file. The user of a rear riser production system does in this way, manufactures PCB using a PCB manufacture data file, and he offers the final example of implementation of an assembly and the input design section for a device and PCB while he constitutes the logic device which cannot be reconfigurated using a binary configuration file. In a rear riser production system, it is not new to use functionally the gate array chip equivalent to ERCGA of the example of implementation of eternal hardware which cannot be reconfigurated, but it is generally carried out. Rather, that this system can make the digital system of the magnitude of arbitration, that this system is expressed with the gestalt of comprehensive base element logic in an input design file (not limited to the logic library of a specific computer maker) (this is not restricted to the capacity of one IC chip), offering the example of implementation of eternal hardware still more nearly automatically, and these can call it one mode of freshness.
[0101] 3.7 A rear riser computing system rear riser hardware system can be constituted according to the motion specified by the input program currently written with a high-class computer language like a pascal. Moreover, according to the memorized program which was memorized for [which a computer performs] being general purpose, a count function can be performed using this. This is attained by using a high-level layout composition compiler, changes a computer program into the gestalt of the digital logic currently expressed in the input design file, and after that, in rear riser hardware, it is operated while it realizes this layout section. This method is a new count means fundamentally. If it sees from the standpoint of count, rear riser hardware will be made into an altitude parallel processing data processor, and it will be the logic function and storage device in the element of a rear riser logic chip, an interconnect chip, and the specific purpose about the data-processing element. This data processor does not necessarily calculate according to the memorized program count method about performing sequential measurement. This data processor is constituted by rear riser hardware, and operates according to the data path which operates according to the motion directed by the input program, a functional unit, and a finite-state machine control structure. This advantage is quicker than a possible count speed in count by the program count speed was sequentially remembered to be. The rear riser computing system to explain is equipped with a rear riser count compiler, rear riser layout section conversion system, and a rear riser count operator with the rear riser hardware system and the host computer (drawing 61). It is cautious of this host computer being used only as a means made to merely run a rear riser count operator, and not being used about performing the count function directed by the input program. other means made to run a rear riser count operator can be used -- it is natural.
[0102] 3.7.1 A rear riser count compiler rear riser count compiler changes into an input design section file the input program file written with the high level language using the text editor. This is equipped with the layout section composition compiler, the logic composition compiler, and the functional unit compiler. A layout section composition compiler is a tool and some of the examples are developed recently ("Tutorial on High-Level Synthesis", McMarland, Parke and Camposano, Proceeding of the 25 th Design Automatic Conference, ACM and IEEE, 1988). This compiler constitutes the description about the finite-state machine controller which consists of functional unit and data I/O and the system of a data path, and bus interconnect, and operates according to the motion specified with a standard procedure computer language. There is "a hula

mel (flamel)" as an example of an actual layout section composition compiler. The method is explained to details by "Flamel:A High-Level Hardware Compiler", Howard Trickey, IEEE Transaction on Computer-Aided Design, Vol.CAD-6, No.2, and 1987. : which shows the citation from reference -- the input to "hula mel is a pascal program. "" user gives the activation frequency in the case of generally performing an input program to a pascal program. Other user inputs are numbers which show what hardware is permitted. An output is layout of the hardware which plays the same role as the PASCAL language. The model made by "" hula mel is a synchronous digital machine which consists of a data path and a controller. The data path consists of the functional units (ALU, an adder, a register, I/O pad, etc.) which interconnect by bus. A controller is a finite-state machine. " -- " -- the motion required of hardware is specified using a general pascal program. The hula mel is enabling implementation of the high-speed activation which agreed with cost constraint according the parallel processing in a program to a header and user assignment. The example of implementation of "" hula mel is completed. An output is description about a data path and a controller. In a series of tests, a hula mel will materialize the program which runs by MC 22 to 200 times the speed of 68000 (microcomputer) although the same program is performed, if a clock cycle is made the same. The input "whose cost constraint specified by "user a user or a rear riser computing system has according to the capacity of the rear riser hardware system used" is supplied to this layout section composition compiler. The output of a layout section composition compiler is a middle expression file equipped with description of a data path and a controller. A functional unit library is the expression of a functional unit with which the lot was defined beforehand. The expression to the functional unit of each type is given by the layout section composition compiler. These expressions specify logic and the device (USD) base elements specified by a user, and these network interconnect. These expressions have agreed with the requirements for a rear riser input design section base element. A USD base element is used additionally and the base element of the higher engine performance or a bigger capacity can be offered rather than what is realized using a logic chip and layout section memory. For example, when attaching a high-speed VLSI floating-point-type multiplier as USD, functional unit binary is equipped with the description about the functional unit of a floating a small number of point type multiplier which specifies this USD base element. A logic composition compiler changes the description about a data path and a finite-state machine controller into the expression about the logic base element and interchange-circuit network in an input count file. This logic composition compiler is equipped with a finite-state machine composition tool. This Mentor Graphics Corp. and VLSI Technology Inc. and Synopsis Inc. etc. ("Logic Synthesis speeds ASIC Design", A.J.de Geus, IEEE Spectrum, and August 1989) Commercially available from -- or It is developed according to a method given in reference (). ["The] Implementation of a State Machine Compiler", C.Kingsley and Proceedings of the 24 th Design Automation Conference, ACM and IEEE, 1987; "A State Machine Synthesizer" and D.Brown, Proceedings of the 18th DesignAutomation Conference, ACM and IEEE and 1981; "An Overview of Logic Synthesis Systems", L.Trevillyan, Proceedings of the 24 th Design Automation Conference, ACM and IEEE, 1987. This compiler reads a middle expression file including the description about :1 data path and collector which operate according to the following methods into a data structure.

2) Change description of the functional unit of each data path into logic, a USD base element, and a network according to description of a functional unit library.

3) Offer the layout section memory base element to each data output from each data input and data path to a data path.

4) Change description of a finite-state machine controller into logic base elements and these network interconnect using a finite-state machine composition tool.

5) Offer the base element of SUTIMYURETA to the 'start' input to a finite-state machine controller, and the 'busy' from finite-state machine Koto L'Ora and the 'Dawn (done)' output, and a sampler.

6) A clock network directs to drive with a rear riser clock generation vessel.

7) Send out a base element and a network to an input design file.

[0103] 3.7.2 A rear riser count operator rear riser count operator constitutes a rear riser system,

and essentially, an input program directs him and he enables activation of the count function realized. A rear riser count operator writes in the file about the output data from a count function while he reads into the configuration file and correspondence table file which are created by layout conversion and reads the file about the input data to the count function specified by a user. In order to operate the count function realized, the configuration file of :1 layout section is read, and using this, as the section of a configuration described, all rear riser logic and interconnect chips are constituted.
2) Read an input data file and write the data in input data layout section memory. Output-data layout section memory is cleared.
3) Read a correspondence table file and determine correspondence between SUTIMYURETA and a sampler and between the host interface bus addresses during a control input and an output.
4) Enable a clock generation machine, assert a 'start' control input through SUTIMYURETA, and make actuation start.
5) When the monitor of the 'Dawn' control output is carried out and this serves as truth, data is read from output design section memory, and this is written in an output data file. In order to use a rear riser computing system, :1 text editor or other means are used, and an input program and an input data file are ******(ed).
2) Generate an input design file using a rear riser count compiler.
3) As others already described, generate a configuration and a correspondence table file using the rear riser layout conversion system which operates by the usual method.
4) Actually perform a count function using a rear riser count operator.
5) Read the data calculated by the realized count function from an output data file.
[0104] 4 Suitable Example Explained through Specification of Suitable Example ** is : [0105] Which Has the Following Features. 4.1 Use hardware partial crossbar interconnect for all hardware systems hierarchical on 3 level. Drawing 62 - drawing 64 shows the general architecture of the logic board which interconnected hierarchical, a box, and a rack. Drawing 65 a-b shows the physical structure about a board, a box, and a rack.
Logic board (drawing 62): Each logic board consists of 14 L chips which interconnected with 32 X level crossbar chip. L chips each have 128 I/O pins for every chip connected to X level crossbar, and four connection has accomplished them to each of 32 X chips. ; using 14 additional I/O pins -- 11 pieces are connected to R bus among those, one piece is connected to each of two clock signals, and one piece is connected to a layout section reset signal. Xilinx XC3090 LCA is used as a logic chip. X chips each are equipped with 56 I/O pins connected to the logic chip, and four connection is made by each of 14 L chips. X chips each are equipped with each of two Y chips, and eight additional I/O pin connections. Xilinx XC2018 LCA is used as an X-chip. Each logic board has 512 back I/O pins to X-Y pass. This also has connection with R bus and a configuration bus.
Box (drawing 63): Each box consists of 1-8 boards which interconnect with 64 Y level crossbar chips. Y chips each have six I/O pins connected to the logic box board, and eight connection has accomplished them for X chip of each board. This has eight additional I/O connections with one Z chip. Xilinx XC2018 LCA is used as a Y chip. 64 Y chips are attached in eight Y chipboard. This each has 512 back I/O pins to X-Y pass. Eight Y chipboard and eight logic boards are interconnected with the wire on the back of X-Y pass of a box. Y chipboard each has 64 I/O pins in the cable connector to Y-Z pass. Each box has such eight connectors. These connection is brought together in 512 single wire Y-Z pass cables from each box. Y chipboard each also has the connection to a configuration bus. Drawing 65 a is equipped with a Y-Z pass cable, and shows the physical configuration on the back of X-Y pass which has a host interface, eight logic boards, and eight Y chipboard.
Rack (drawing 64): Each rack is equipped with 1-8 boxes, and interconnects with 64 Z level crossbar chip. Z chips each are equipped with 64 I/O pins connected to the box, and eight connection is made by Y chip of each box. Xilinx XC2018 LCA is used as a Z chip. The box of a rack is interconnected with an additional box using connection with the X-Z pass cable from each box arranged at the logic board. The physical configuration of Z level box is shown in drawing 65 b. 64 Z chips are attached in eight Z chipboard. This each has 512 I/O pins to Y-Z

pass. Eight Z chipboard and eight Y-Z pass gable connectors are interconnected by trace on the back of Y-Z pass. As the section of memory described, the memory module respectively equipped with 16 RAM chips and ten LCA is attached in the location of the logic chip which is the location for which it is needed. The memory module is used for layout section memory, vector memory, SUTIMYURETA, and a sampler as prescribed by the section of a stimulus and a response. The hardware device module by user assignment is attached in the location of the logic chip LCA. One certain box is equipped with the host interface board which is carrying out cable splicing to the I/O-bus interface card of a host computer. This box controls the host interface bus called R bus. This bus is connected to configuration control logic block in all logic chip locations, each logic board, i.e., Y chipboard, and Z chipboard for all control and data transmission functions. R bus is equipped with a 8-bit data path, a clock, and two control lines as the section described. The host interface board is equipped with a configuration bus controller, two clock generation machines, and reset controllers. The configuration bus which has a 16-bit data path connects all logic and crossbar chips to all configuration functions using a host interface. 14 L chips of each board are made into one configuration group, and the 32 X chip is divided into two groups. Let eight Y chipboard of each box be one group respectively like each of eight Z chipboard.

[0106] 4.2 software design sections conversion system is :Quick Sim indicated with the section consist of the following modules and concerning [the each] each. It has a logic base element. Layout section reader which reads a Mentor Graphics layout file. Quick Sim Base element converter which changes a base element into a Xilinx LCA base element. Tri-state and a wye yard network driver are changed according to a crossbar addition configuration, as the section of tri-state described. Party SHONA based on cluster composition technology as the section described. The interconnect and the network listing system which send out the netlist file of the XNF format about each logic and the crossbar chip in a system while interconnecting the partial crossbar of three level. Xilinx which consists of XNF2LCA, APR, and Makebits LCA netlist conversion tool. Configuration file collector.

** ** Mentor Graphics Rear riser logic simulation system which uses the RSIM batch interfacing tool based on the log file. Mentor Graphics Rear riser fault simulation system which uses the RSIM batch simulation interface tool based on the log file. Mentor Graphics Quick Sim Rear riser logic simulator evaluation system which acts as a logic simulator. The rear riser prototyping system which is equipped with a logic analyzer and has the virtual meter realized.

Rear riser executive system Mentor Graphics Board Station Automatic PCB arrangement and rear riser production system using a routing tool. The rear riser computing system using the PASCAL language, a hula mel layout section composition compiler, and Mentor Graphics Design, Knowledge and Logic Consultant FSM and a logic composition tool. although the suitable example was quoted and the principle of this invention was explained, it separates with such a principle and various equipments and details can be changed -- it is clear. For example, Mentor Graphics Although it explained that this invention operated effectively using the modification of electrical-design automation, it turns out that this invention can be similarly carried out using other layout section automation tools. Deformation and modification various by within the limits which is not limited to the example currently indicated here and does not change a summary are possible for this invention.

[Translation done.]